Keywords Forms

versie 3.0, 1993

Inhoudsopgave

Voorwoord

SDW-Forms biedt de gebruiker de mogelijkheid op eenvoudige en overzichtelijke wijze de Systeem Encyclopedie te presenteren en te bewerken.

SE benaderen

De standaard-wijze om in SDW de SE te benaderen (via het SE-icon), kent slechts twee mogelijkheden:

- precies één rubriek openen en bewerken
- alle rubrieken van een component in een BROWSE-overzicht tonen

Met SDW-Forms is het mogelijk elke gewenste combinatie van rubrieken van een component in een window zichtbaar te maken en te bewerken. Daarbij kan niet alleen de selectie en volgorde van die rubrieken door de gebruiker bepaald worden, maar kan de gebruiker ook mede bepalen op welke wijze de rubriek gepresenteerd wordt.

De enige beperking is dat een SDW-formulier altijd wordt gedefinieerd voor maximaal één component-type en dat binnen een formulier altijd slechts één component tegelijk getoond en bewerkt kan worden. Het is derhalve niet mogelijk in één formulier tegelijkertijd rubrieken van verschillende componenten of component-typen zichtbaar te maken of te bewerken.

Formulieren definiëren

De inhoud en vormgeving van elk SDW-formulier wordt vast gelegd in een formulier-definitie. Dat is een ASCII-file die bestaat uit Formulier-keywords en bijbehorende argumenten. Deze ASCII-files met als extensie .FRM kunnen naar keuze worden bewaard in de volgende directories:

- systeemdirectories
- LIBWS-directory
- FORMS-directory
- LIB-directory

Als een SDW-formulier wordt geopend, wordt de bijbehorende definitie regel voor regel verwerkt en wordt het formulier in een window opgebouwd.

↖ Uitgebreide beschrijving SDW-formulieren
SDWorkstation 2: Forms

Alfabetisch overzicht Formulier-keywords

In dit overzicht worden alle keywords beschreven die gebruikt kunnen worden bij het definiëren van SDW-formulieren.

De beschrijving van elk keyword bestaat uit de volgende onderdelen:

- de syntax van het keyword
- een korte aanduiding van de functie van het keyword
- een uitgebreide beschrijving van de mogelijkheden en beperkingen van het keyword
- een verwijzing naar verwante keywords

Wat de verschillende onderdelen inhouden, komt hierna aan de orde.

Syntax

Aangegeven wordt welke parameters bij het keyword horen en van welke aard deze parameters dienen te zijn.
Daarbij worden de volgende regels gevolgd:

< x >	=	parameter x is verplicht.
< x | y >	=	parameter verplicht, keuze uit parameter x of y.
[x]	=	parameter x is optioneel.
[x | y]	=	parameter optioneel, keuze uit parameter x of y.
[x [y]]	=	beide parameters optioneel, parameter y kan echter alleen voorkomen als ook parameter x voorkomt en is dan niet verplicht.
{ x }	=	herhaling, d.w.z. 0 (nul) of meer malen de parameter x.
{ x | y }	=	herhaling, d.w.z. 0 (nul) of meer malen een van de parameters x en y of een combinatie van beide.

Functie

Korte aanduiding van de functie van het betreffende keyword, met tussen haakjes een verwijzing naar het type keyword. Er wordt een onderscheid gemaakt tussen de volgende type SDW-formulier-keywords:

- SDW — deze keywords hebben een direct verband met de SDW-Systeem Encyclopedie (component-typen, componenten, rubrieken)
- Lay out — deze keywords zijn van belang voor de vormgeving van een formulier
- Pos — deze keywords betreffende positionering binnen een formulier
- Button — deze keywords houden verband met in een formulier op te nemen buttons
- Form — deze keywords hebben een direct verband met het aanroepen en definiëren van een formulier
- Script — deze keywords betreffen het gebruik van SDWrite-scripts in combinatie met formulieren
- \- — deze keywords zijn van een ander type dan de hiervoor beschrevene

Voorbeeld:

(MDFORM)
Functie: na een linkerklik op een MDFORM-button wordt het betreffende formulier geladen. (Button/Form)

Werking

Uitgebreide beschrijving van de mogelijkheden van het betreffende keyword, met informatie over mogelijke toepassingen. Eventueel worden een of meer voorbeelden gegeven. Ook wordt aangegeven welke vereisten en beperkingen bij het betreffende keyword gelden.

Zie ook

Verwijzing naar de belangrijkste verwante Formulier-keywords, dat wil zeggen keywords die een soortgelijke functie hebben als het betreffende keyword of die bij het gebruik van het keyword een belangrijke rol spelen.

BOX

Syntax: BOX <breedte | $breedte> <hoogte | $hoogte>

Functie: plaatsen van een rechthoek. (Lay Out)

Werking: Hiermee wordt een rechthoek getekend vanaf de actieve kolom en rij met de opgegeven breedte en hoogte, òf (indien geen argumenten zijn meegegeven) de breedte en hoogte die aan het laatste **SIZE**-commando is meegegeven.

Een box is géén button: de getekende rechthoek heeft geen enkele andere functie dan het verduidelijken van de formulier-lay out.
Het is mogelijk in de box teksten, plaatjes en buttons op te nemen. Ook in dat geval is de box echter niets anders dan een figuur op het formulier.
Bijvoorbeeld:

```
SIZE 11 3
BOX
DOWN 1
RIGHT 1
TEXT IN DE BOX
```

Hierdoor wordt een box geplaatst, waarin de tekst 'IN DE BOX' zichtbaar zal zijn. De box is daarmee echter geen button geworden.

Indien slechts één argument wordt meegegeven, wordt dit als breedte genomen. De hoogte is dan 1.

Zie ook: **LINE, SIZE.**

BROWSE

Syntax: BROWSE

Functie: aangeven dat een geheel formulier of een veld binnen het formulier niet gemuteerd mag worden. (-)

Werking: Indien het keyword **BROWSE** bovenaan in een formulier-definitie voorkomt, dat wil zeggen vóór het **TYPE**-commando, wordt daarmee aangegeven dat geen van de in dat formulier voorkomende velden in het formulier gemuteerd kan worden. Het gehele formulier wordt dan read only getoond.
Indien het gaat om een formulier zònder **TYPE**-commando, dient het **BROWSE**-commando te worden opgenomen voor het eerste veld in het formulier.

Bijvoorbeeld:

```
DESCRIPTION Gegevenselementen
BROWSE
TYPE Gegevenselement
```

Indien **BROWSE** wordt opgenomen direct voor een regel met **ITEM**, **FILE**, **FORMDEF**, **SCRIPTDEF** of **VAR** dan wordt slechts dat veld in BROWSE-mode geopend. Andere velden in het formulier kunnen in dat geval wèl gemuteerd worden.
Bijvoorbeeld:

```
TEXT BESCHRIJVING
BROWSE
ITEM Beschrijving
```

Zie ook: -.

BUTTONSIZE

Syntax: BUTTONSIZE <breedte | $breedte>

Functie: opgeven van de breedte van buttons. (Forms)

Werking: Het keyword **BUTTONSIZE** is bedoeld om een de breedte van buttons in te stellen en werkt onafhankelijk van het keyword **SIZE**.
Elke button die na een **BUTTONSIZE**-commando wordt gedefinieerd, zal de aangegeven breedte hebben, totdat met behulp van een nieuw **BUTTONSIZE**-commando een nieuwe button-breedte wordt ingesteld.
De hoogte van een button wordt automatisch op 1 ingesteld.

Bij het plaatsen van een button wordt de bij die button meegegeven tekst gecentreerd, uitgaande van de bij **BUTTONSIZE** ingestelde breedte.

Zie ook: **SIZE**.

COLUMN

Syntax: COLUMN <kolomnummer | $kolomnummer>

Functie: positioneren op de aangegeven kolom van de huidige regel. (Pos)

Werking: Het keyword **COLUMN** is bedoeld om een **absolute** kolompositie in te stellen. De regelpositie blijft hierbij ongewijzigd. Elk formulier-veld dat gedefinieerd wordt na het **COLUMN**-commando zal op de aangegeven kolom worden gepositioneerd, totdat met behulp van een nieuw positionerings-commando een nieuwe kolompositie wordt ingesteld.

Indien een **relatieve** verplaatsing naar een andere kolompositie gewenst is, kan gebruik gemaakt worden van de commando's **LEFT** en **RIGHT**.

Het verschil tussen de commando-reeksen:

```
COLUMN 3
COLUMN 5
```

en

```
COLUMN 3
RIGHT 5
```

is derhalve dat de actieve kolom in het eerste geval kolom 5 zal zijn en in het tweede geval kolom 8.

Zie ook: **LEFT**, **RIGHT**, **ROW**, **UP**, **DOWN**, **POP** en **PUSH**.

COMMENT

Syntax: COMMENT [tekst]

Functie: toevoegen van commentaar aan een formulier-definitie. (-)

Werking: Elke regel in een formulier-definitie die begint met **COMMENT** wordt beschouwd als commentaar. Op de verwerking van het formulier hebben dergelijke regels geen invloed.

Door commentaar aan een formulier-definitie toe te voegen, wordt de leesbaarheid en onderhoudbaarheid verbeterd. Om die reden wordt aangeraden elke formulier-definitie van het noodzakelijke commentaar te voorzien.

Zie ook: -.

COMPONENT

Syntax: COMPONENT <componentnaam | $componentnaam>

Functie: selecteren van een default-component voor het formulier. (SDW)

Werking: Bij het openen van een formulier voor een bepaald component-type wordt standaard gevraagd om de component waarvoor het formulier geopend dient te worden. Hiertoe verschijnt een selectie-window waaruit de gewenste naam geselecteerd kan worden.

Indien het formulier echter altijd voor dezelfde component geopend dient te worden, kan dit aangegeven worden door een **COMPONENT**-commando in de formulier-definitie op te nemen. Bij het openen van het formulier zal dan direct de bij die component behorende formulier-inhoud getoond worden.
Met behulp van de buttons **FIRST**, **PREVIOUS**, **NEXT**, **LAST** en **SELECT** kan vervolgens vanuit het formulier eventueel een andere component actief gemaakt worden.

Zie ook: **FIRST**, **PREVIOUS**, **NEXT**, **LAST**, **SELECT**.

DESCRIPTION

Syntax: DESCRIPTION <omschrijving>

Functie: korte beschrijving van het formulier. (Form)

Werking: Een selectie-window van SDW-componenten bevat standaard uitsluitend de namen van de betreffende componenten. Bij SDW-formulieren en SDWrite-scripts kan hieraan voor elke naam ook een korte omschrijving van het formulier/SDWrite-script worden toegevoegd door in de definitie een **DESCRIPTION** op te nemen.

De tekst achter **DESCRIPTION** wordt in selectie-windows tussen vierkante haken achter de component-naam getoond. Bovendien verschijnt de tekst in de naamrand van het formulier, eveneens tussen vierkante haken.

Bijvoorbeeld: indien in de definitie van het SDW-formulier 'formdef' het volgende commando wordt opgenomen:

```
DESCRIPTION Formulieren ontwerpen
```

zal dit formulier in selectie-windows als volgt getoond worden:

```
formdef [Formulieren ontwerpen]
```

Aangeraden wordt elke definitie van een SDW-formulier en SDWrite-script van een **DESCRIPTION**-regel te voorzien.

Dit commando **moet** overigens als eerste commando in het formulier staan, dat wil zeggen nog **voor** het **TYPE**-commando.

Zie ook: **COMMENT**.

DOWN

Syntax: DOWN [aantal | $aantal]

Functie: het aangegeven aantal regels naar beneden gaan. (Pos)

Werking: Het keyword **DOWN** is bedoeld om een **relatieve** regelpositie in te stellen, gerekend vanaf de op dat moment actieve regel. De kolompositie blijft hierbij ongewijzigd.
Elk formulier-veld dat gedefinieerd wordt na het **DOWN**-commando zal op de aangegeven regel worden gepositioneerd, totdat met behulp van een nieuw positionerings-commando een nieuwe regelpositie wordt ingesteld.

Indien een **absolute** verplaatsing naar een andere regelpositie gewenst is, kan gebruik gemaakt worden van het commando **ROW**.

Het verschil tussen de commando-reeksen:

```
ROW 3
ROW 5
```

en

```
ROW 3
DOWN 5
```

is derhalve dat de actieve regel in het eerste geval regel 5 zal zijn en in het tweede geval regel 8.

Indien **geen** waarde achter **DOWN** wordt opgegeven, is de actieve **SIZE**-hoogte bepalend voor het aantal regels dat gedaald wordt. Vergelijk de volgende commando-reeksen:

```
SIZE 70 5
ITEM Beschrijving
DOWN 6
```

en

```
SIZE 70 5
ITEM Beschrijving
DOWN
DOWN 1
```

Indien wordt besloten de hoogte van het **ITEM**-veld te vergroten (bijvoorbeeld van 5 naar 7), zal in het eerste geval ook het **DOWN**-commando aangepast moeten worden (van 6 naar 8). In het tweede geval is dit niet nodig: dankzij het commando **DOWN** zonder argument wordt dan immers automatisch reeds 7 regels gedaald. Het extra commando om nog een regel te dalen, zorgt in beide gevallen voor de gewenste regel tussenruimte.
Aangeraden wordt dan ook om zo vaak mogelijk het **DOWN**-commando zonder argument te gebruiken, aangezien het onderhoud van de formulieren hiermee wordt vereenvoudigd, met name als er veel velden in een formulier voorkomen.

Zie ook: **ROW**, **UP**, **SIZE**.

FILE

Syntax: FILE <filenaam | $filenaam> [SMALL]

Functie: voor het tonen van een ASCII-file. (-)

Werking: In een **FILE**-veld wordt de inhoud getoond van een ASCII-file. De file kan vanuit het formulier bewerkt worden, tenzij het **FILE**-commando wordt voorafgegaan door het **BROWSE**-commando.

De filenaam dient compleet met eventueel pad te worden opgegeven.

Standaard verschijnt aan de rechterzijde van het **FILE**-veld een '+'-symbool. Na een linkerklik op dit symbool wordt het veld getoond op de volledige window-grootte.
Indien het niet gewenst is dat het '+'-symbool verschijnt, dient als tweede argument 'SMALL' te worden opgenomen. Het veld kan in dat geval niet tot window-grootte worden uitvergroot.

★ Een veld van het type **FILE** is niet hetzelfde als een rubriek-veld voor een rubriek van het type externe file!

Zie ook: **BROWSE**.

FIRST

Syntax: FIRST [tekst | $tekst]

Functie: actief maken van de eerste component van het geselecteerde type. (SDW, Button)

Werking: Een **FIRST**-button dient voor het selecteren van de alfabetisch eerste component van het component-type waarvoor het formulier is gedefinieerd (met behulp van het keyword **TYPE**). Na een linkerklik op een **FIRST**-button wordt het formulier opnieuw getekend, met in de velden de invullingen voor de eerste component.

In de button wordt de naam getoond die als argument achter **FIRST** is opgegeven. Deze tekst wordt binnen de button gecentreerd.
Indien geen naam wordt aangegeven achter het keyword wordt de button in het binnen SDW bepaalde standaardformaat getekend.

★ Om de grootte van de button in te stellen kan gebruik gemaakt worden van het commando **BUTTONSIZE**.

Zie ook: **LAST**, **NEXT**, **PREVIOUS** en **TYPE**.

FORMDEF

Syntax: FORMDEF <formnaam | $formnaam> [SMALL]

Functie: definiëren van een formulier. (Form)

Werking: In een **FORMDEF**-veld kan een formulier-definitie worden aangemaakt of gewijzigd. Het opgegeven formulier wordt hierbij gezocht volgens het standaard-zoekpad.

Indien een formulier nog niet voorkomt, wordt de ingevoerde definitie bewaard in de systeem-directory.
Indien het formulier reeds bestaat in een van de directories in het standaard-zoekpad, worden eventuele wijzigingen bewaard in de directory waar het formulier gevonden is.

In alle gevallen wordt de ingetoetste definitie bewaard als ASCII-file met als extensie .FRM en als naam de opgegeven formuliernaam.

Standaard verschijnt aan de rechterzijde van het **FORMDEF**-veld een '+'-symbool. Na een linkerklik op dit symbool wordt het veld getoond op de volledige window-grootte.
Indien het niet gewenst is dat het '+'-symbool verschijnt, dient als tweede argument 'SMALL' te worden opgenomen. Het veld kan in dat geval niet tot window-grootte worden uitvergroot.

Er dient hierbij rekening gehouden te worden met het volgende: de opgegeven formuliernaam kan maximaal 32 tekens lang zijn, de naam van de ASCII-file is echter maximaal 8 tekens lang. Indien twee formulieren worden gedefinieerd waarvan de eerste 8 tekens van de naam hetzelfde zijn, zal een van beide formulieren derhalve verloren gaan.
Zorg daarom voor formuliernamen waarvan de eerste 8 tekens uniek zijn.

Zo zullen beide commando's:

```
FORMDEF klantgegevens_invoeren
```

betrekking hebben op een ASCII-file met de naam 'klantgeg.frm'.

Houdt er tevens rekening mee dat de componentnaam een naam moet zijn die geldig is als MS-DOS-filenaam !!!
Wordt een foutieve naam opgegeven dan kan en zal de ingebrachte tekst niet bewaard worden !!!!

Zie ook: **SCRIPTDEF, MDFORM.**

Standaard-zoekpad
SDWorkstation 1: Zoekvolgorde

GENERATED_ITEM

Syntax: GENERATED_ITEM <naam | $naam> [SMALL]

Functie: tonen van de inhoud van een gegenereerde rubriek. (SDW)

Werking: Met behulp van **GENERATED_ITEM** wordt aangegeven dat in het formulier een veld geopend dient te worden waarin de inhoud van de aangegeven gegenereerde rubriek wordt getoond. Bij het starten van het formulier wordt voor de actieve component automatisch de inhoud van de gegeneerde rubriek in het veld geplaatst.

Daarbij verschijnt standaard aan de rechterzijde een '+'-symbool. Na een linkerklik op dit symbool wordt het veld getoond op de volledige window-grootte.
Indien het niet gewenst is dat het '+'-symbool verschijnt, dient als tweede argument 'SMALL' te worden opgenomen. Het veld kan in dat geval niet tot window-grootte worden uitvergroot.

Gegenereerde rubrieken kunnen **niet** bewerkt worden.

Zie ook: -.

HEADER

Syntax: HEADER <naam | $naam>

Functie: tonen van de header van de (gegenereerde) rubriek. (SDW)

Werking: Met behulp van **HEADER** wordt de header van de rubriek of gegenereerde rubriek in een formulier geplaatst. Dit is een tekstveld, waarmee geen verdere bewerkingen kunnen worden uitgevoerd.
De header van een 'gewone' rubriek kan worden ingesteld in het SDW-hulpprogramma Onderhoud SE. De header van een gegenereerde rubriek is altijd de naam van die gegenereerde rubriek, maar dan volledig in hoofdletters.

De header van een rubriek kan in beginsel ook met behulp van een **TEXT**-commando in een formulier worden opgenomen. Indien bijvoorbeeld voor de rubriek 'beschrijving' als header 'BESCHRIJVING' is opgegeven, dan leiden de volgende twee commando's tot hetzelfde resultaat:

```
HEADER beschrijving
```

en

```
TEXT BESCHRIJVING
```

Het voordeel van het **HEADER**-commando is echter dat bij een wijziging van de rubrieksheader met behulp van Onderhoud SE geen wijzigingen in de formulier-definitie noodzakelijk zijn.

★ Indien het gewenst is dat bij een **ITEM**-veld de rubrieksheader van de betreffende rubriek wordt getoond, wordt daarom aangeraden hiervoor het **HEADER**-commando te gebruiken en niet het **TEXT**-commando.

★ Vanzelfsprekend dient de opgegeven rubrieksnaam **exact** overeen te komen met de rubrieksnaam die is opgegeven in Onderhoud SE (let op hoofdletters/kleine letters/speciale tekens

!!!).

Indien een naam wordt opgegeven die noch als 'gewone' rubriek noch als gegenereerde rubriek bestaat, wordt de achter **HEADER** geplaatste tekst in hoofdletters afgedrukt.

Zie ook: **ITEM**, **GENERATED_ITEM**.

↖ Definiëren van een rubrieksheader
SDWorkstation 2: Onderhoud SE

IMAGE

Syntax: IMAGE <imagefile>

Functie: tonen van een plaatje in een formulier. (Lay Out)

Werking: In een formulier kunnen plaatjes worden opgenomen in pcx-formaat (MS-DOS) of bmp-formaat (MS-DOS of MS-Windows). Deze plaatjes worden gezocht volgens het standaard zoekpad.
Een **IMAGE**-plaatje is uitsluitend bedoeld als verfraaiing van een formulier, een linkerklik in een **IMAGE**-plaatje heeft geen acties tot gevolg. Indien het wel gewenst is dat een linkerklik in een plaatje leidt tot een actie dient gebruik gemaakt te worden van het keyword **PICTURE**.

De plaatjes worden getekend zonder kader. Indien een kader gewenst is, dient hiertoe een **BOX** getekend te worden.

De grootte van de afbeelding hoeft niet overeen te komen met de actieve **SIZE**. De afbeelding wordt automatisch gecentreerd en indien nodig afgekapt.

Zie ook: **PICTURE**, **BOX**.

↖ Standaard-zoekpad
SDWorkstation 1: Zoekvolgorde

INCLUDE

Syntax: INCLUDE <formnaam>

Functie: opnemen van een ander formulier in het huidige formulier. (Form)

Werking: Indien in verschillende formulieren gebruik wordt gemaakt van een aantal overeenkomstige velden, hoeven deze velden niet in elke formulier-definitie opnieuw te worden aangegeven. In dat geval kan volstaan worden met een **INCLUDE**-commando waarin wordt aangegeven in welk formulier de algemene velden staan.

★ Aangezien een formulier altijd voor ten hoogste één component-type gedefinieerd kan worden, wordt aangeraden om in de sub-formulieren die met **INCLUDE** worden opgenomen **geen** **TYPE**-commando op te nemen.

Een formulier-definitie die met **INCLUDE** wordt opgenomen, wordt verwerkt alsof die formulier-definitie op de betreffende plaats in het aanroepende formulier is opgenomen.

Ook het **INCLUDE**-formulier wordt gezocht volgens het standaard zoekpad.

Zie ook: **TYPE**, **FORMDEF**.

 Standaard-zoekpad
SDWorkstation 1: Zoekvolgorde

ITEM

Syntax: ITEM <naam | $naam> [SMALL]

Functie: tonen van de inhoud van een rubriek. (SDW)

Werking: Met behulp van **ITEM** wordt aangegeven dat een rubriek-veld geopend dient te worden. Hierin wordt voor de actieve component de inhoud getoond van de rubriek met de opgegeven rubrieksnaam.

★ Vanzelfsprekend dient de opgegeven rubrieksnaam **exact** overeen te komen met de rubrieksnaam die is opgegeven in Onderhoud SE (let op hoofdletters/kleine letters/speciale tekens !!!).

Afhankelijk van het type van de rubriek zijn in het kader de gebruikelijke bewerkingen mogelijk, tenzij de regel met **ITEM** direct wordt voorafgegaan door een regel met het keyword **BROWSE**. In dat geval is het niet mogelijk de inhoud van de rubriek aan te passen.

Standaard verschijnt aan de rechterzijde van het **ITEM**-veld een '+'-symbool. Na een linkerklik op dit symbool wordt het veld getoond op de volledige window-grootte.
Indien het niet gewenst is dat het '+'-symbool verschijnt, dient als tweede argument 'SMALL' te worden opgenomen. Het veld kan in dat geval niet tot window-grootte worden uitvergroot.

Zie ook: **BROWSE**.

LAST

Syntax: LAST [naam | $naam]

Functie: actief maken van de laatste component van het geselecteerde type. (SDW, Button)

Werking: Een **LAST**-button dient voor het selecteren van de alfabetisch laatste component van het component-type waarvoor het formulier is gedefinieerd (met behulp van het keyword **TYPE**). Na een linkerklik op een **LAST**-button wordt het formulier opnieuw getekend, met in de velden de invullingen voor de laatste component.

In de button wordt de naam getoond die als argument achter **LAST** is opgegeven. Deze tekst wordt binnen de button gecentreerd.
Indien geen naam wordt aangegeven achter het keyword wordt de button in het binnen SDW bepaalde standaardformaat getekend.

Om de grootte van de button in te stellen kan gebruik gemaakt worden van het commando **BUTTONSIZE**.

Zie ook: **FIRST**, **NEXT**, **PREVIOUS** en **TYPE**.

LEFT

Syntax: LEFT [aantal | $naam]

Functie: het aangegeven aantal kolommen naar links gaan. (Pos)

Werking: Het keyword **LEFT** is bedoeld om een **relatieve** kolompositie in te stellen, gerekend vanaf de op dat moment actieve kolom. De regelpositie blijft hierbij ongewijzigd.
Elk formulier-veld dat gedefinieerd wordt na het **LEFT**-commando zal op de aangegeven kolom worden gepositioneerd, totdat met behulp van een nieuw positioneringscommando een nieuwe kolompositie wordt ingesteld.

Indien een **absolute** verplaatsing naar een andere kolompositie gewenst is, kan gebruik gemaakt worden van het commando **COLUMN**. Het verschil tussen de commando-reeksen:

```
COLUMN 10
COLUMN 3
```

en

```
COLUMN 10
LEFT 3
```

is derhalve dat de actieve kolom in het eerste geval kolom 3 zal zijn en in het tweede geval kolom 7.

Indien **geen** waarde achter **LEFT** wordt opgegeven, is de actieve **SIZE**-breedte bepalend voor het aantal kolommen dat naar links wordt gegaan. Bijvoorbeeld:

```
SIZE 32 1
ITEM Lengte
LEFT
```

In dit geval zal 32 posities naar links gegaan worden.

Zie ook: **COLUMN, RIGHT, UP, DOWN, SIZE.**

LINE

Syntax: LINE <breedte | $breedte> <hoogte | $hoogte>

Functie: tekenen van een lijn. (Lay Out)

Werking: Met behulp van het commando **LINE** kunnen lijnen getekend worden.
De lengte en richting van de lijn worden bepaald door de opgegeven breedte en hoogte, òf (indien geen argumenten zijn meegegeven) door de breedte en hoogte die aan het laatste **SIZE**-commando is meegegeven.
Een lijn met een actieve breedte of lengte van 0 loopt horizontaal respectievelijk verticaal. Alle andere lijnen lopen diagonaal in het formulier.

Bijvoorbeeld: indien de actieve positie de derde regel, kolom 1 is, zullen de volgende commando's:

```
LINE 0 30
LINE 30 0
LINE 30 30
```

leiden tot drie lijnen: een verticale lijn in kolom 1, een horizontale lijn op regel 3 en een lijn die diagonaal 30 posities naar rechts en 30 regels naar beneden loopt.

Zie ook: **IMAGE**.

MDFORM

Syntax: MDFORM <tekst | $tekst> <formnaam | $formnaam> [pagina] [component]

Functie: een button voor het starten van een ander formulier. (Form)

Werking: Met behulp van het keyword **MDFORM** kan vanuit een formulier een ander formulier worden geladen. Daartoe wordt een button in het formulier geplaatst met de meegegeven tekst. Een linkerklik op de button zorgt er vervolgens voor dat het aangegeven formulier wordt geopend.

Door bijvoorbeeld in de formulier-definitie van formulier X de volgende opdracht op te nemen:

```
MDFORM SYSINFO systeem
```

wordt in het gedefinieerde formulier X een button getekend met als opschrift 'SYSINFO'. Indien in formulier X een linkerklik wordt gegeven op de button 'SYSINFO' zal het formulier 'systeem' gestart worden.

Indien een formulier uit meerdere pagina's bestaat, kan tevens worden aangegeven dat een bepaalde **pagina** van het aangeroepen formulier dient te worden getoond. Hiertoe wordt als derde argument het betreffende pagina-nummer opgenomen.
Bijvoorbeeld:

```
MDFORM SYSINFO systeem 2
```

Hiermee wordt na een linkerklik op de button 'SYSINFO' pagina 2 van formulier 'systeem' geopend.

Bij het openen van een formulier voor een bepaald component-type wordt standaard gevraagd om de component waarvoor het formulier geopend dient te worden. Hiertoe verschijnt een selectie-window waaruit de gewenste naam geselecteerd kan worden.

Indien het aangeroepen formulier echter altijd voor dezelfde component geopend dient te worden, kan dit aangegeven worden door als vierde argument de naam van de gewenste component op te nemen. Bij het openen van het formulier zal dan direct de bij die component behorende formulier-inhoud getoond worden. Deze functionaliteit is gelijk aan die van het keyword **COMPONENT**; dit laatste keyword heeft echter betrekking op het actieve formulier, terwijl het vierde argument achter **MDFORM** de component bepaalt waarvoor het **aangeroepen** formulier geopend dient te worden.
Bijvoorbeeld:

```
MDFORM SYSINFO systeem 2 autoexec
```

Hiermee wordt na een linkerklik op de button 'SYSINFO' pagina 2 van formulier 'systeem' geopend voor de component 'autoexec'.

Merk op dat bij gebruik van het vierde argument ook altijd het derde argument gevuld dient te zijn. Indien het aangeroepen formulier slechts uit 1 pagina bestaat, dient als derde argument een '1' ingevuld te worden. Bijvoorbeeld:

```
MDFORM SYSINFO systeem 1 config
```

Zie ook: **COMPONENT**.

NEXT

Syntax: NEXT [naam | $naam]

Functie: actief maken van de volgende component van het geselecteerde type. (SDW, Button)

Werking: Een **NEXT**-button dient voor het selecteren van de alfabetisch volgende component van het component-type waarvoor het formulier is gedefinieerd (met behulp van het keyword **TYPE**). Na een linkerklik op een **NEXT**-button wordt het formulier opnieuw getekend, met in de velden de invullingen voor de volgende component.

In de button wordt de naam getoond die als argument achter **NEXT** is opgegeven. Deze tekst wordt binnen de button gecentreerd.
Indien geen naam wordt aangegeven achter het keyword wordt de button in het binnen SDW bepaalde standaardformaat getekend.

★ Om de grootte van de button in te stellen kan gebruik gemaakt worden van het commando **BUTTONSIZE**.

Zie ook: **FIRST**, **LAST**, **PREVIOUS** en **TYPE**.

OK

Syntax: OK [tekst | $tekst]

Functie: sluiten van het formulier. (Form)

Werking: Met behulp van het keyword **OK** wordt een button getekend met de aangegeven tekst. Een linkerklik op een **OK**-button zorgt ervoor dat het formulier-window gesloten wordt.

In de button wordt de naam getoond die als argument achter **OK** is opgegeven. Deze tekst wordt binnen de button gecentreerd.
Indien geen naam wordt aangegeven achter het keyword wordt de button in het binnen SDW bepaalde standaardformaat getekend.

★ Om de grootte van de button in te stellen kan gebruik gemaakt worden van het commando **BUTTONSIZE**.

Zie ook: -.

PAGE

Syntax: PAGE [nummer | $nummer]

Functie: aangeven dat de volgende velden op een andere formulier-pagina voorkomen. (Form)

Werking: Een formulier-definitie kan zo uitgebreid gemaakt worden als de gebruiker wenst. Het is mogelijk dat het formulier daarbij groter wordt dan de maximale (scherm-)grootte. In dat geval zullen niet alle velden op het scherm zichtbaar gemaakt worden.
Door in zo'n formulier-definitie een of meer keren het commando **PAGE** op te nemen, wordt het formulier opgebouwd uit verschillende pagina's die elk afzonderlijk wel op het scherm kunnen passen.
Met behulp van de keywords **PAGE_UP**, **PAGE_DOWN** en **PAGE_NR** kan vervolgens eenvoudig binnen het formulier genavigeerd worden.

Er is een duidelijk **verschil tussen het gebruik van pagina's en het gebruik van verschillende formulieren**: de verschillende pagina's worden gedefinieerd binnen 1 formulier-definitie, terwijl verschillende formulieren overeenkomen met even zoveel formulier-definities.

Zie ook: **PAGE_DOWN, PAGE_UP, PAGE_NR.**

PAGE_DOWN

Syntax: PAGE_DOWN [tekst | $tekst]

Functie: tonen van een button waarmee de volgende formulier-pagina wordt geladen. (Button, Form)

Werking: Een formulier-definitie kan zo uitgebreid gemaakt worden als de gebruiker wenst. Het is mogelijk dat het formulier daarbij groter wordt dan de maximale (scherm-)grootte. In dat geval zullen niet alle velden op het scherm zichtbaar gemaakt worden.
Door in zo'n formulier-definitie een of meer keren het commando **PAGE** op te nemen, wordt het formulier opgebouwd uit verschillende pagina's die elk afzonderlijk wel op het scherm kunnen passen.
Met behulp van de keywords **PAGE_UP**, **PAGE_DOWN** en **PAGE_NR** kan vervolgens eenvoudig binnen het formulier genavigeerd worden.

Met behulp van het keyword **PAGE_DOWN** wordt een button getekend met de tekst die als argument is opgegeven. Deze tekst wordt binnen de button gecentreerd.
Indien geen tekst wordt aangegeven, wordt de button in het binnen SDW bepaalde standaardformaat getekend.

Een linkerklik op de button zorgt ervoor dat de volgende pagina van het formulier wordt geladen.

Om de grootte van de button in te stellen kan gebruik gemaakt worden van het commando **BUTTONSIZE**.

Zie ook: **PAGE, PAGE_UP, PAGE_NR.**

PAGE_NR

Syntax: PAGE_NR <tekst | $tekst> <nummer | $nummer>

Functie: tonen van een button waarmee de aangegeven formulier-pagina wordt geladen. (Button, Form)

Werking: Een formulier-definitie kan zo uitgebreid gemaakt worden als de gebruiker wenst. Het is mogelijk dat het formulier daarbij groter wordt dan de maximale (scherm-)grootte. In dat geval zullen niet alle velden op het scherm zichtbaar gemaakt worden.
Door in zo'n formulier-definitie een of meer keren het commando **PAGE** op te nemen, wordt het formulier opgebouwd uit verschillende pagina's die elk afzonderlijk wel op het scherm kunnen passen.
Met behulp van de keywords **PAGE_UP**, **PAGE_DOWN** en **PAGE_NR** kan vervolgens eenvoudig binnen het formulier genavigeerd worden.

Met behulp van het keyword **PAGE_NR** wordt een button getekend met de tekst die als argument is opgegeven. Deze tekst wordt binnen de button gecentreerd.
Indien geen tekst wordt aangegeven, wordt de button in het binnen SDW bepaalde standaardformaat getekend.

Een linkerklik op de button zorgt ervoor dat de aangegeven pagina van het formulier wordt geladen.

★ Om de grootte van de button in te stellen kan gebruik gemaakt worden van het commando **BUTTONSIZE**.

Zie ook: **PAGE_DOWN**, **PAGE_UP**, **PAGE**.

PAGE_UP

Syntax: PAGE_UP [tekst | $tekst]

Functie: tonen van een button waarmee de vorige formulier-pagina wordt geladen. (Button, Form)

Werking: Een formulier-definitie kan zo uitgebreid gemaakt worden als de gebruiker wenst. Het is mogelijk dat het formulier daarbij groter wordt dan de maximale (scherm-)grootte. In dat geval zullen niet alle velden op het scherm zichtbaar gemaakt worden.
Door in zo'n formulier-definitie een of meer keren het commando **PAGE** op te nemen, wordt het formulier opgebouwd uit verschillende pagina's die elk afzonderlijk wel op het scherm kunnen passen.
Met behulp van de keywords **PAGE_UP**, **PAGE_DOWN** en **PAGE_NR** kan vervolgens eenvoudig binnen het formulier genavigeerd worden.

Met behulp van het keyword **PAGE_UP** wordt een button getekend met de tekst die als argument is opgegeven. Deze tekst wordt binnen de button gecentreerd.
Indien geen tekst wordt aangegeven, wordt de button in het binnen SDW bepaalde standaardformaat getekend.

Een linkerklik op de button zorgt ervoor dat de vorige pagina van het formulier wordt geladen.

Om de grootte van de button in te stellen kan gebruik gemaakt worden van het commando **BUTTONSIZE**.

Zie ook: **PAGE_DOWN, PAGE_NR, PAGE.**

PICTURE

Syntax: PICTURE <button-keyword> <plaatje>

Functie: aangeven dat een button niet met een tekst maar met een plaatje aangegeven dient te worden. (Button, Lay Out)

Werking: Elk button-commando (zoals **OK**, **FIRST**, **LAST** en **MDFORM**) leidt standaard tot het verschijnen van een button met een standaard-tekst of een door de gebruiker aangegeven tekst. Indien het echter gewenst is dat een button wordt getoond in de vorm van een plaatje, dient het keyword **PICTURE** voor het betreffende button-commando te worden gezet. Tevens dient in dat geval de naam van het gewenste plaatje te worden aangegeven.
In een formulier kunnen plaatjes worden opgenomen in pcx-formaat (MS-DOS) of bmp-formaat (MS-DOS of MS-Windows). Deze plaatjes worden gezocht volgens het standaard zoekpad.

De grootte van de afbeelding hoeft niet overeen te komen met de actieve **SIZE**. De afbeelding wordt automatisch gecentreerd en indien nodig afgekapt.

Een linkerklik op een **PICTURE**-button heeft dezelfde gevolgen als een linkerklik op een 'gewone' button.
Bijvoorbeeld:

```
OK ja
PICTURE OK ja
```

Beide commando's leiden tot het plaatsen van een **OK**-button in het formulier. In het eerste geval zal deze button de tekst 'ja' bevatten, in het tweede geval zal volgens het standaard-zoekpad een plaatje met de naam ja.pcx of ja.bmp gezocht worden. Dit plaatje zal vervolgens in het formulier getoond worden en de functionaliteit van een 'gewone' **OK**-button hebben.

Zie ook: **FIRST**, **IMAGE**, **LAST**, **MDFORM**, **NEXT**, **OK**, **PAGE_DOWN**,

PAGE_NR, PAGE_UP, PREVIOUS, SCRIPT, SELECT.

Standaard-zoekpad
SDWorkstation 1: Zoekvolgorde

POP

Syntax: POP

Functie: het positioneren op de positie die met het vorige **PUSH**-commando is opgeslagen. (Pos)

Werking: Om het positioneren van velden in een formulier te vereenvoudigen, kan gebruik gemaakt worden van de keywords **PUSH** en **POP**.
Met **PUSH** worden de huidige positie en **SIZE** opgeslagen; met **POP** wordt gepositioneerd op de positie van de laatste **PUSH** en wordt de opgeslagen **SIZE** weer actief gemaakt. De keywords werken hierbij volgens het Last In First Out-principe.
Bijvoorbeeld:

```
POS 3 1
SIZE 8 3
PUSH
POS 15 1
SIZE 7 7
PUSH
POS 20 1
POP
POP
```

In dit geval zal de eerste **POP** positie 15 1 en **SIZE** 7 7 activeren; de tweede **POP** maakt positie 3 1 en **SIZE** 8 3 actief.

Het gebruik van **POP** en **PUSH** maakt formulieren eenvoudiger onderhoudbaar. Het wijzigen van een positie wordt immers 'doorgerekend' in de gebruikte **POP**-commando's.
Indien bijvoorbeeld drie kolommen geschreven worden, kan de **PUSH/POP**-combinatie gebruikt worden om de kolommen naast elkaar te zetten:

```
POS 1 1
SIZE 10 10
PUSH
TEXT kolom 1
[...]
POP             (positioneert op 1 1 en zet size op 10 10)
```

```
RIGHT          (positioneert 10 posities naar rechts, omdat de
               size-breedte 10 is)
RIGHT 2        (nog twee posities naar rechts)
PUSH           (nieuwe positie 13 1 en size 10 10 opslaan)
TEXT kolom 2
[...]
POP            (positioneert op 13 1 en zet size op 10 10)
RIGHT          (positioneert 10 posities naar rechts, omdat de
               size-breedte 10 is)
RIGHT 2        (nog twee posities naar rechts)
TEXT kolom 3
[...]
```

In dit geval is slechts eenmaal een 'harde' positie gezet (met **POS** 1 1) en worden alle overige posities berekend. Wijziging van de uitgangspositie in bijvoorbeeld 3 1 leidt nu automatisch tot het fraai meeschuiven van de overige teksten.

Zie ook: **PUSH, POS, LEFT, RIGHT, UP, DOWN.**

POS

Syntax: POS <kolom | $kolom> <rij | $rij>

Functie: positioneren op de aangegeven positie. (Pos)

Werking: Met behulp van een **POS**-commando wordt de aangegeven positie (regel en kolom) actief gemaakt. Hierna gedefinieerde velden of buttons zullen op de aangegeven positie worden geplaatst.

Met **POS** wordt een **absolute** positionering uitgevoerd ten opzichte van de linkerbovenhoek van het formulier-window. Indien in een formulier veel van dit commando gebruik wordt gemaakt en er is een verplaatsing noodzakelijk, dan zullen vaak **alle** **POS**-commando's aangepast moeten worden.

Aangeraden wordt het **POS**-commando zo weinig mogelijk te gebruiken. De onderhoudbaarheid van een formulier wordt aanmerkelijk vergroot door zo vaak mogelijk gebruik te maken van commando's die zorgen voor een **relatieve** positionering, zoals **LEFT**, **RIGHT**, **UP**, **DOWN**, **POP** en **PUSH**.

Zie ook: **LEFT**, **RIGHT**, **UP**, **DOWN**, **POP**, **PUSH**.

POST_SCRIPT

Syntax: POST_SCRIPT <scriptnaam>

Functie: uitvoeren van een SDWrite-script bij het sluiten van een formulier. (Script, Form)

Werking: Indien bij het sluiten van een formulier bepaalde handelingen dienen te worden uitgevoerd, bijvoorbeeld het bijwerken van rapporten op basis van de ingevoerde gegevens of het wijzigen van bepaalde variabelen, kan aan het betreffende formulier een SDWrite-script gekoppeld worden dat bij het afsluiten wordt uitgevoerd. Dit SDWrite-script wordt een post-script genoemd en wordt uitgevoerd indien in het formulier het **POST_SCRIPT**-commando is opgenomen.

Het betreffende SDWrite-script wordt gezocht volgens het standaard-zoekpad. De extensie '.SCR' hoeft niet te worden aangegeven.

★ Er kan per formulier slechts één **POST_SCRIPT**-script worden aangegeven.

Zie ook: **PRE_SCRIPT, VAR.**

↖ Standaard-zoekpad
SDWorkstation 1: Zoekvolgorde

PRE_SCRIPT

Syntax: PRE_SCRIPT <scriptnaam>

Functie: uitvoeren van een SDWrite-script bij het openen van een formulier. (Script, Form)

Werking: Indien bij het openen van een formulier bepaalde handelingen dienen te worden uitgevoerd, bijvoorbeeld het opslaan van de huidige inhoud van het te openen formulier of het initialiseren van bepaalde variabelen, kan aan het betreffende formulier een SDWrite-script gekoppeld worden dat bij het openen wordt uitgevoerd. Dit SDWrite-script wordt een pre-script genoemd en wordt uitgevoerd indien in het formulier het **PRE_SCRIPT**-commando is opgenomen.

Het betreffende SDWrite-script wordt gezocht volgens het standaard-zoekpad. De extensie '.SCR' hoeft niet te worden aangegeven.

★ Er kunnen per formulier één of meer **PRE_SCRIPT**-scripts worden aangegeven. Deze worden na elkaar uitgevoerd.

Zie ook: **POST_SCRIPT**, **VAR**.

↖ Standaard-zoekpad
SDWorkstation 1: Zoekvolgorde

PREVIOUS

Syntax: PREVIOUS [tekst | $tekst]

Functie: actief maken van de vorige component van het geselecteerde type. (SDW, Button)

Werking: Een **PREVIOUS**-button dient voor het selecteren van de alfabetisch vorige component van het component-type waarvoor het formulier is gedefinieerd (met behulp van het keyword **TYPE**). Na een linkerklik op een **PREVIOUS**-button wordt het formulier opnieuw getekend, met in de velden de invullingen voor de vorige component.

In de button wordt de naam getoond die als argument achter **PREVIOUS** is opgegeven. Deze tekst wordt binnen de button gecentreerd.
Indien geen naam wordt aangegeven achter het keyword wordt de button in het binnen SDW bepaalde standaardformaat getekend.

★ Om de grootte van de button in te stellen kan gebruik gemaakt worden van het commando **BUTTONSIZE**.

Zie ook: **FIRST**, **LAST**, **NEXT** en **TYPE**.

PUSH

Syntax: PUSH

Functie: het opslaan van de huidige positie en **SIZE** ten behoeve van een **POP**-commando. (Pos)

Werking: Om het positioneren van velden in een formulier te vereenvoudigen, kan gebruik gemaakt worden van de keywords **PUSH** en **POP**.
Met **PUSH** worden de huidige positie en **SIZE** opgeslagen; met **POP** wordt gepositioneerd op de positie van de laatste **PUSH** en wordt de opgeslagen **SIZE** weer actief gemaakt. De keywords werken hierbij volgens het Last In First Out-principe.
Bijvoorbeeld:

```
POS 3 1
SIZE 8 3
PUSH
POS 15 1
SIZE 7 7
PUSH
POS 20 1
POP
POP
```

In dit geval zal de eerste **PUSH** positie 3 1 en **SIZE** 8 3 opslaan; de tweede **PUSH** slaat positie 15 1 en **SIZE** 7 7 op. De twee **POP**-commando's maken positie 15 1 en **SIZE** 7 7 (eerste **POP**) en positie 3 1 en **SIZE** 8 3 (tweede **POP**) actief.

Het gebruik van **POP** en **PUSH** maakt formulieren eenvoudiger onderhoudbaar. Het wijzigen van een positie wordt immers 'doorgerekend' in de gebruikte **POP**-commando's.
Indien bijvoorbeeld drie kolommen geschreven worden, kan de **PUSH/POP**-combinatie gebruikt worden om de kolommen naast elkaar te zetten:

```
POS 1 1
SIZE 10 10
PUSH
TEXT kolom 1
```

```
[...]
POP             (positioneert op 1 1 en zet size op 10 10)
RIGHT           (positioneert 10 posities naar rechts, omdat de
                size-breedte 10 is)
RIGHT 2         (nog twee posities naar rechts)
PUSH            (nieuwe positie 13 1 en size 10 10 opslaan)
TEXT kolom 2
[...]
POP             (positioneert op 13 1 en zet size op 10 10)
RIGHT           (positioneert 10 posities naar rechts, omdat de
                size-breedte 10 is)
RIGHT 2         (nog twee posities naar rechts)
TEXT kolom 3
[...]
```

In dit geval is slechts eenmaal een 'harde' positie gezet (met **POS** 1 1) en worden alle overige posities berekend. Wijziging van de uitgangspositie in bijvoorbeeld 3 1 leidt nu automatisch tot het fraai meeschuiven van de overige teksten.

Zie ook: **POP, POS, LEFT, RIGHT, UP, DOWN.**

RIGHT

Syntax: RIGHT [tekst | $tekst]

Functie: het aangegeven aantal kolommen naar rechts gaan. (Pos)

Werking: Het keyword **RIGHT** is bedoeld om een **relatieve** kolompositie in te stellen, gerekend vanaf de op dat moment actieve kolom. De regelpositie blijft hierbij ongewijzigd.
Elk formulier-veld dat gedefinieerd wordt na het **RIGHT**-commando zal op de aangegeven kolom worden gepositioneerd, totdat met behulp van een nieuw positioneringscommando een nieuwe kolompositie wordt ingesteld.

Indien een **absolute** verplaatsing naar een andere kolompositie gewenst is, kan gebruik gemaakt worden van het commando **COLUMN**. Het verschil tussen de commando-reeksen:

```
COLUMN 5
COLUMN 8
```

en

```
COLUMN 5
RIGHT 8
```

is derhalve dat de actieve kolom in het eerste geval kolom 8 zal zijn en in het tweede geval kolom 13.

Indien **geen** waarde achter **RIGHT** wordt opgegeven, is de actieve **SIZE**-breedte bepalend voor het aantal kolommen dat naar rechts wordt gegaan. Bijvoorbeeld:

```
SIZE 32 1
ITEM Lengte
RIGHT
```

In dit geval zal 32 posities naar rechts gegaan worden.

Zie ook: **COLUMN, LEFT, UP, DOWN, SIZE.**

ROW

Syntax: ROW <rijnummer | $rijnummer>

Functie: positioneren op de aangegeven regel en de huidige kolom. (Pos)

Werking: Het keyword **ROW** is bedoeld om een **absolute** regelpositie in te stellen. De kolompositie blijft hierbij ongewijzigd.
Elk formulier-veld dat gedefinieerd wordt na het **ROW**-commando zal op de aangegeven regel worden gepositioneerd, totdat met behulp van een nieuw positionerings-commando een nieuwe regelpositie wordt ingesteld.

Indien een **relatieve** verplaatsing naar een andere regelpositie gewenst is, kan gebruik gemaakt worden van de commando's **UP** en **DOWN**.

Het verschil tussen de commando-reeksen:

```
ROW 3
ROW 5
```

en

```
ROW 3
DOWN 5
```

is derhalve dat de actieve regel in het eerste geval regel 5 zal zijn en in het tweede geval regel 8.

Zie ook: **UP**, **DOWN**, **LEFT**, **RIGHT**, **COLUMN**, **POP** en **PUSH**.

SCRIPT

Syntax: SCRIPT <tekst | $tekst> <scriptnaam>

Functie: plaatsen van een button met de aangegeven tekst waarmee het aangegeven SDWrite-script gestart wordt. (Script, Button)

Werking: Door in een formulier een **SCRIPT**-button op te nemen, kan vanuit dat formulier een SDWrite-script gestart worden.
In de button wordt de tekst getoond die als argument achter **SCRIPT** is opgegeven. Deze tekst wordt binnen de button gecentreerd.

Het SDWrite-script wordt gezocht via het standaard-zoekpad. De extensie '.SCR' hoeft niet te worden aangegeven.

Met behulp van **SCRIPT**-buttons kunnen formulieren ook gebruikt worden als uitgangspunt voor rapportage-doeleinden.

★ Om de grootte van de button in te stellen kan gebruik gemaakt worden van het commando **BUTTONSIZE**.

Zie ook: **PRE_SCRIPT**, **POST_SCRIPT**, **SCRIPTDEF**.

↖ Standaard-zoekpad
SDWorkstation 1: Zoekvolgorde

SCRIPTDEF

Syntax: SCRIPTDEF <scriptnaam | $scriptnaam> [SMALL]

Functie: definiëren van een SDWrite-script. (Script)

Werking: In een **SCRIPTDEF**-veld kan een script-definitie worden aangemaakt of gewijzigd. Het opgegeven SDWrite-script wordt hierbij gezocht volgens het standaard-zoekpad.

Indien een SDWrite-script nog niet voorkomt, wordt de ingevoerde definitie bewaard in de systeem-directory.
Indien het SDWrite-script reeds bestaat in een van de directories in het standaard-zoekpad, worden eventuele wijzigingen bewaard in de directory waar het SDWrite-script gevonden is.

In alle gevallen wordt de ingetoetste definitie bewaard als ASCII-file met als extensie .SCR en als naam de opgegeven scriptnaam.

Standaard verschijnt aan de rechterzijde van het **SCRIPTDEF**-veld een '+'-symbool. Na een linkerklik op dit symbool wordt het veld getoond op de volledige window-grootte.
Indien het niet gewenst is dat het '+'-symbool verschijnt, dient als tweede argument 'SMALL' te worden opgenomen. Het veld kan in dat geval niet tot window-grootte worden uitvergroot.

Er dient hierbij rekening gehouden te worden met het volgende: de opgegeven scriptnaam kan maximaal 32 tekens lang zijn, de naam van de ASCII-file is echter maximaal 8 tekens lang.
Indien twee SDWrite-scripts worden gedefinieerd waarvan de eerste 8 tekens van de naam hetzelfde zijn, zal een van beide SDWrite-scripts derhalve verloren gaan.
Zorg daarom voor scriptnamen waarvan de eerste 8 tekens uniek zijn.

Zo zullen beide commando's:

SCRIPTDEF klantgegevens_invoeren
SCRIPTDEF klantgegevens_wijzigen

betrekking hebben op een ASCII-file met de naam 'klantgeg.scr'.

☞ Houdt er tevens rekening mee dat de componentnaam een naam moet zijn die geldig is als MS-DOS-filenaam !!!
Wordt een foutieve naam opgegeven dan kan en zal de ingebrachte tekst niet bewaard worden !!!!

Zie ook: **FORMDEF**.

↖ Standaard-zoekpad
SDWorkstation 1: Zoekvolgorde

SELECT

Syntax: SELECT [tekst | $tekst]

Functie: actief maken van de gewenste component van het geselecteerde type. (SDW, Button)

Werking: Een **SELECT**-button dient voor het selecteren van de gewenste component van het component-type waarvoor het formulier is gedefinieerd (met behulp van het keyword **TYPE**). Na een linkerklik op een **SELECT**-button verschijnt een selectie-window met componentnamen waaruit de gewenste naam gekozen kan worden of een nieuwe naam kan worden ingevoerd. Vervolgens wordt het formulier opnieuw getekend, met in de velden de invullingen voor de geselecteerde component.

In de button wordt de naam getoond die als argument achter **SELECT** is opgegeven. Deze tekst wordt binnen de button gecentreerd.
Indien geen naam wordt aangegeven achter het keyword wordt de button in het binnen SDW bepaalde standaardformaat getekend.

★ Om de grootte van de button in te stellen kan gebruik gemaakt worden van het commando **BUTTONSIZE**.

Zie ook: **FIRST**, **NEXT**, **PREVIOUS**, **LAST** en **TYPE**.

↖ Naam invoeren of selecteren uit selectie-window
SDWorkstation 1: Selectie-window

SIZE

Syntax: SIZE <breedte | $breedte> <hoogte | $hoogte>

Functie: instellen van de hoogte en breedte van een te tekenen veld. (Lay Out)

Werking: Voor elk veld in een formulier dient aangegeven te worden met welke afmetingen dat veld getekend dient te worden. Deze gewenste grootte kan worden ingesteld met behulp van het **SIZE**-commando. Hiermee worden de breedte (in kolommen) en de hoogte (in rijen) gedefinieerd.
Indien slechts één argument wordt opgegeven, wordt daarmee de breedte ingesteld. De hoogte is standaard 1.

De aangegeven **SIZE** wordt gebruikt voor alle erna gedefinieerde velden tot er een nieuwe **SIZE** wordt ingesteld.
Bijvoorbeeld:

```
SIZE 70 5
ITEM beschrijving
[...]
ITEM commentaar
[...]
SIZE 32 1
ITEM lengte
```

In dit geval worden de **ITEM**-velden voor de rubrieken 'beschrijving' en 'commentaar' beide met een grootte van 70 bij 5 getoond. Het veld voor de rubriek 'lengte' wordt getekend met een grootte van 32 bij 1.

★ Het gebruik van de combinatie **POP** en **PUSH** kan leiden tot een aanpassing van de **SIZE**. Met **PUSH** wordt namelijk de **SIZE** ingesteld die actief was op het moment van het bijbehorende **POP**-commando.
Bijvoorbeeld:

```
SIZE 30 10
POP
SIZE 20 10
PUSH
```

In dit geval zal het veld voor de rubriek commentaar getoond worden met een grootte van 30 bij 10, aangezien die grootte actief was bij het commando **POP**. Het commando **PUSH** roept die opgeslagen grootte weer op, waardoor de ingestelde **SIZE** van 20 10 ongedaan wordt gemaakt.

Om de grootte van een **button** in te stellen kan gebruik gemaakt worden van het commando **BUTTONSIZE**. Dit commando werkt onafhankelijk van het **SIZE**-commando.

Zie ook: **POP**, **PUSH**.

SPACING

Syntax: SPACING <aantal | $aantal>

Functie: aanpassen van de regel-spatiëring. (Lay Out)

Werking: Indien twee velden in een formulier direct onder elkaar worden geplaatst, zullen de kaderranden elkaar standaard overlappen. Om deze overlapping te voorkomen, kan aangegeven worden dat gebruik gemaakt dient te worden van een extra spatiëring.

★ Deze spatiëring kan slechts eenmaal per formulier worden ingesteld en geldt voor het gehele formulier.

Zie ook: -.

SWAP

Syntax: SWAP <NORMAL | DELETE >

Functie: bepaalt of een formulier-window bij het openen van een ander formulier vanuit dat formulier-window geopend dient te blijven (NORMAL) of gesloten dient te worden (DELETE). (Form)

Werking: Indien vanuit een formulier een ander formulier wordt aangeroepen (bijvoorbeeld met **MDFORM**), blijft het eerste formulier standaard geopend. Indien frequent andere formulieren worden geopend, zullen derhalve steeds meer windows geopend zijn.

Indien het gewenst is dat het formulier **gesloten** wordt bij het openen van een ander formulier, dient in de formulier-definitie het volgende commando te worden opgenomen:

```
SWAP DELETE
```

Bij een aanroep van een ander formulier, zal het oorspronkelijke formulier in dit geval gesloten worden.

Is het gewenst dat het oorspronkelijke window geopend blijft (de normale situatie), dan kan het volgende commando worden opgenomen:

```
SWAP NORMAL
```

Aangezien dit de standaard-instelling is, is dit commando echter niet noodzakelijk.

Zie ook: **MDFORM**, **OK**.

TEXT

Syntax: text <tekst | $tekst>

Functie: plaatsen van tekst in een formulier. (Lay Out)

Werking: Om een formulier te verduidelijken, kunnen tekstregels worden toegevoegd op alle gewenste posities. Hiermee kan bijvoorbeeld een identificerend commentaar bij een veld geplaatst worden. Te denken valt hierbij met name aan velden voor het tonen en bewerken van (gegenereerde) rubrieken, files en variabelen. Door daarboven of ernaast aan te geven om welke rubriek, file of variabele het gaat, wordt een formulier aanmerkelijk overzichtelijker.
Bijvoorbeeld:

```
TEXT BESCHRIJVING
DOWN 2
SIZE 75 10
ITEM beschrijving
```

Deze commando-reeks toont een veld voor de rubriek 'beschrijving' met een grootte van 75 bij 10. Boven het veld wordt de tekst 'BESCHRIJVING' getoond.

★ Indien de aangegeven tekst **variabelen** bevat, worden deze op een van de volgende twee manieren getoond:

- in de tekst wordt de waarde getoond die de variabele bij het openen van het formulier had
- in de tekst wordt de naam van de variabele getoond als deze variabele bij het openen van het formulier nog geen waarde had

★ Indien het gewenst is dat de variabelen in een formulier voortdurend de actuele inhoud tonen, dient gebruik gemaakt te worden van het commando **VAR**.

Zie ook: **SIZE, VAR.**

TYPE

Syntax: TYPE <component-type>

Functie: bepalen van het component-type waarvoor het formulier wordt gedefinieerd. (SDW)

Werking: Formulieren kunnen gebruikt worden om gegevens over componenten in te voeren. Daarbij geldt dat voor zo'n formulier aangegeven dient te worden voor welk component-type het formulier is gedefinieerd.
Indien een formulier bijvoorbeeld gebruikt wordt om gegevens over componenten van het type Functie in te voeren, dient de formulier-definitie de volgende regel te bevatten:

```
TYPE Functie
```

Het is **niet** mogelijk om in één formulier gegevens in te voeren voor verschillende componenten of eenzelfde formulier te gebruiken voor verschillende component-types.

Het is daarentegen **wel** toegestaan om een formulier te ontwerpen dat geen verband houdt met componenten en derhalve evenmin met een component-type. Een dergelijk formulier hoeft dan geen **TYPE**-commando te bevatten.
Formulieren zonder **TYPE**-commando kunnen bijvoorbeeld gebruikt worden als sub-formulier, die met **INCLUDE** in andere formulieren worden ingelezen.

★ Een formulier bevat derhalve altijd nul of één maal het **TYPE**-commando.

Zie ook: **COMPONENT, INCLUDE.**

UP

Syntax: UP [aantal | $aantal]

Functie: het aangegeven aantal regels naar boven gaan. (Pos)

Werking: Het keyword **UP** is bedoeld om een **relatieve** regelpositie in te stellen, gerekend vanaf de op dat moment actieve regel. De kolompositie blijft hierbij ongewijzigd.
Elk formulier-veld dat gedefinieerd wordt na het **UP**-commando zal op de aangegeven regel worden gepositioneerd, totdat met behulp van een nieuw positionerings-commando een nieuwe regelpositie wordt ingesteld.

Indien een **absolute** verplaatsing naar een andere regelpositie gewenst is, kan gebruik gemaakt worden van het commando **ROW**.

Het verschil tussen de commando-reeksen:

```
ROW 5
ROW 3
```

en

```
ROW 5
UP 3
```

is derhalve dat de actieve regel in het eerste geval regel 3 zal zijn en in het tweede geval regel 2.

Indien **geen** waarde achter **UP** wordt opgegeven, is de actieve **SIZE**-hoogte bepalend voor het aantal regels dat gestegen wordt. Vergelijk de volgende commando-reeksen:

```
SIZE 70 5
ITEM Beschrijving
UP 5
```

en

```
SIZE 70 5
ITEM Beschrijving
UP
```

Beide commandoreeksen leiden tot een stijging van 5 regels. Indien wordt besloten de hoogte van het **ITEM**-veld te vergroten (bijvoorbeeld van 5 naar 7), zal in het eerste geval echter nog steeds 5 regels gestegen worden, terwijl in het tweede geval dan automatisch 7 regels wordt gestegen.

Zie ook: **ROW**, **DOWN**, **SIZE**.

VAR

Syntax: VAR <$variabele-naam> [NUMERIC]

Functie: plaatsen van een veld waarin de waarde van de aangegeven variabele kan worden ingevoerd. (Script, Form)

Werking: In een formulier kan gebruikt gemaakt worden van variabelen. Deze variabelen kunnen van een waarde worden voorzien met behulp van SDWrite-scripts (met name pre-scripts), maar het is ook mogelijk de gewenste waarde in te voeren in een **VAR**-veld.

Indien bijvoorbeeld de variabele $aantal dient te worden ingesteld vanuit een formulier, dient in de definitie van dat formulier het volgende commando te worden opgenomen:

```
VAR $aantal
```

In dit geval is het bovendien aannemelijk dat het gewenst is dat uitsluitend een getal als waarde wordt ingetoetst. Dit kan afgedwongen worden door het extra argument 'NUMERIC' op te nemen. Het commando ziet er dan als volgt uit:

```
VAR $aantal NUMERIC
```

Het is nu niet mogelijk in het veld iets anders dan een getal in te voeren.

Zie ook: **PRE_SCRIPT, POST_SCRIPT, SCRIPT.**