



CD351

Advanced ABAP Programming

Contributing Speakers

Ralph Benzinger
Developer, SAP

Thomas Jung
Product Manager, SAP

Björn Mielenhausen
Development Manager, SAP

As a result of this workshop, you will be able to:

- Deploy **strings** for efficient handling of text-based information
- Harness **regular expressions** to process strings effectively
- Utilize **data references** for type-agnostic programming
- Work proficiently with **internal tables**
- Employ **dynamic statements** in your reports

- Benefit from new **ABAP features of NetWeaver 2004s!**



Strings and Regular Expressions

Data References

Internal Tables

Dynamic Statements

Sequential Data Types in ABAP

Business information = sequences of bytes or characters

- Customer names, addresses
- Product numbers, dates, currencies
- XML data
- Natural language

Overview of sequential data types in ABAP

	Character	Special	Byte
Fixed Size	C	D, T, N	X
Variable Size	STRING CSEQUENCE	CLIKE	XSTRING

Comparison of Strings and Fixed-Sized Fields

Strings	Fields
Length adjusted dynamically at runtime	Length fixed and preallocated
Max length of 2,000,000,000	Max length of 65,000
Trailing blanks are significant	Trailing blanks are ignored
Backquote literals <code>`stringlit`</code>	Straight quote literals <code>'fieldlit'</code>
Substring idiom <code>+offset (length)</code> for read access only	Subfield idiom <code>+offset (length)</code> for read and write access

Some **benefits** that ABAP strings provide

- Efficient **management** of string storage space
- Automatic **garbage collection**
- Automatic **value sharing** with copy-on-write

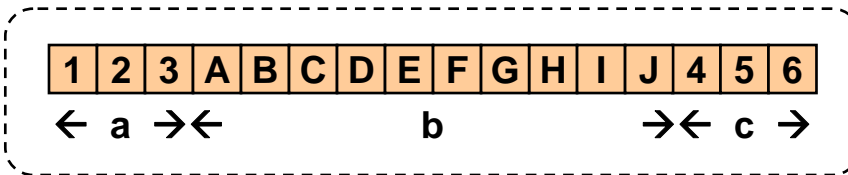
Strings Values

String values are stored separately from variables in memory

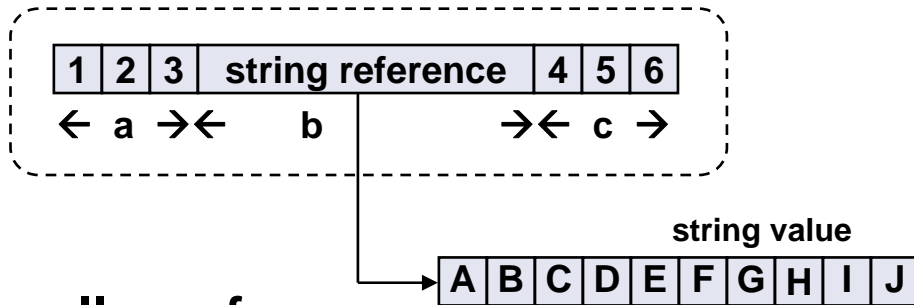
```
DATA: BEGIN OF struc1,
      a(3) TYPE n,
      b(10) TYPE c,
      c(3) TYPE n,
END OF struc1.
```

```
DATA: BEGIN OF struc2,
      a(3) TYPE n,
      b TYPE string,
      c(3) TYPE n,
END OF struc2.
```

struc1 is flat



struc2 is deep



Decoupling of value and reference allows for

- Efficient resizing of strings
- Sharing of string values

String Storage Allocation

Memory allocation during a typical program run

```
DATA str TYPE string.
```

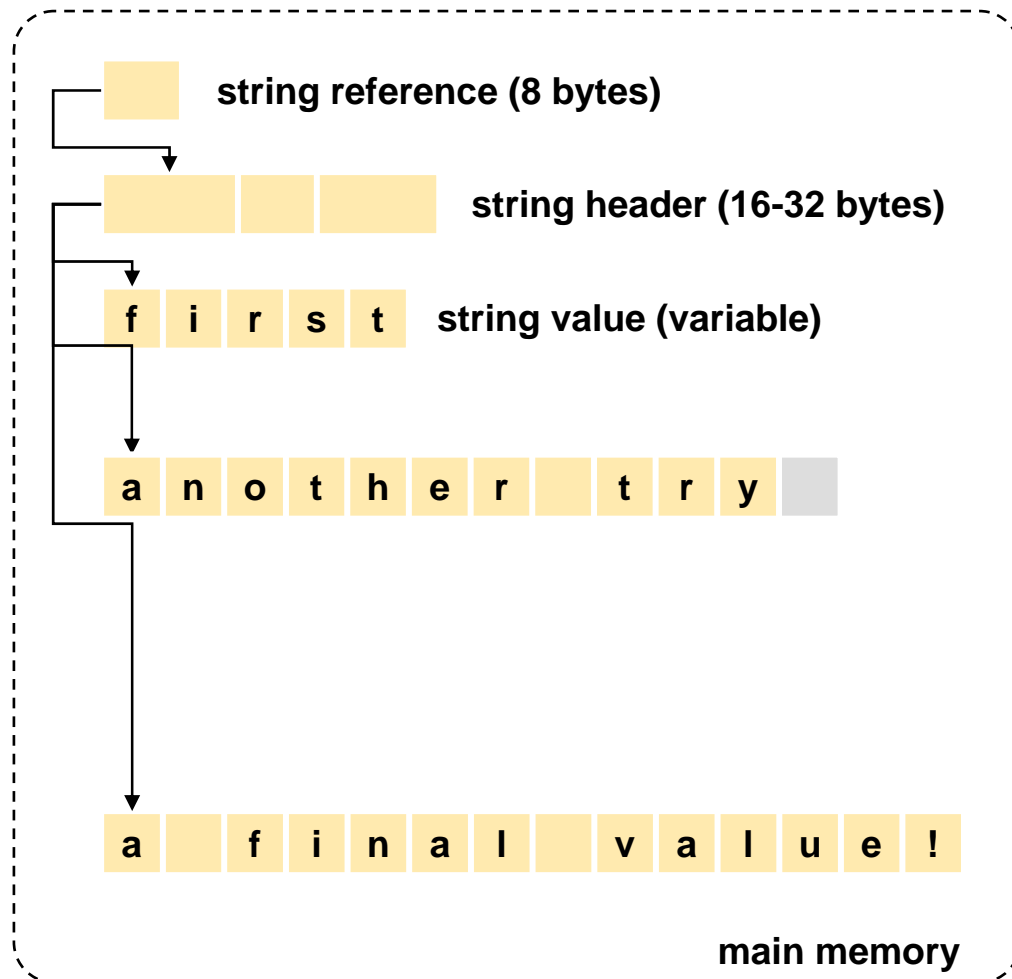
```
str = 'first'.
```

```
str = 'second value'.
```

```
SHIFT str BY 5 PLACES.
```

```
str = 'another try'.
```

```
str = 'a final value!'.
```



Values may be shared by multiple string references

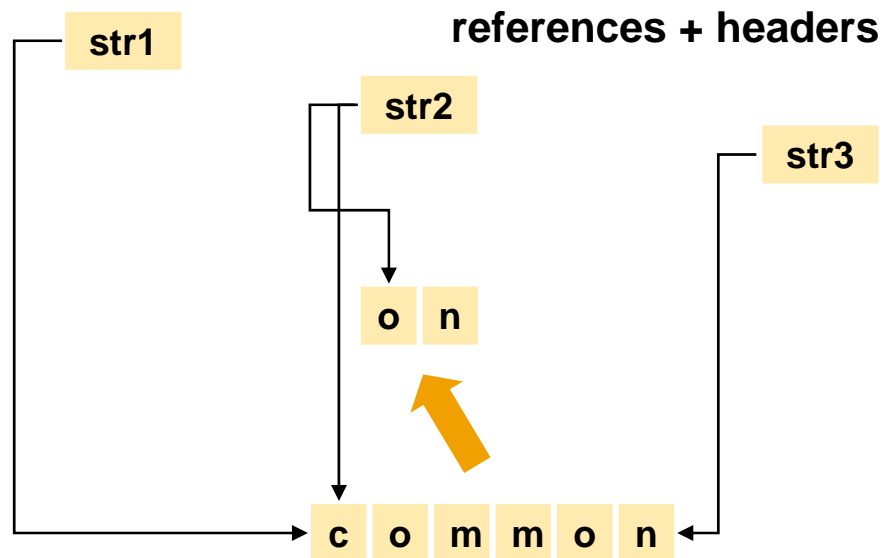
```
DATA: str1 TYPE string,  
      str2 TYPE string,  
      str3 TYPE string.
```

```
str1 = 'common'.
```

```
str2 = str1.
```

```
str3 = str2.
```

```
SHIFT str2 BY 2 PLACES.
```



Unsharing → copy-on-write / lazy copy

- Delays costly copying as long as possible

Restrictions on Substrings

+offset (length) cannot be used to modify strings

```
DATA str TYPE string VALUE `strvalue`.
```

```
WRITE / str+3(4).      " OK
str+3(4) = `x`.  " would modify value
```

- Use REPLACE instead

```
REPLACE SECTION OFFSET 3 LENGTH 4 OF str WITH `x`.
```

Field symbols cannot be used for substring access

- Value changes would necessitate updating all assigned field symbols

```
FIELD-SYMBOLS <fs> TYPE any.
```

```
ASSIGN str+3(4) TO <fs>.
SHIFT str RIGHT BY 2 PLACES.  " shifts char positions
```

Which Data Type to Choose

Advantages of **character fields**

- Easy to access (+off (len), flat structures)
- No internal overhead for accessing value

Advantages of **strings**

- Storage allocated dynamically

```
DATA custname TYPE c LENGTH ??.  
READ DATASET dset INTO line.
```

- Common values are shared

```
DATA customers TYPE TABLE OF cust_record.  
" cust_record = ( name, street, city )
```

- Trailing blanks are significant (may increase performance)

```
DATA c80 TYPE c LENGTH 80 VALUE 'hello'.  
CONCATENATE c80 'world' INTO c80.
```

Checks 75 trailing
blanks for finding
end of text

Display list of largest strings

Goto ->

Display condition ->

Memory use

The screenshot shows the SAP ABAP Debugger interface. The 'Goto' menu item in the top navigation bar is circled in yellow. Below the menu bar, the main program is identified as 'ZJANZH_TEMP'. The source code area shows the following code:

```
EVENT START-OF-SELECTION
do 1000 times.
    buffer = sy-index.
```

The 'Memory Use - Ranked List' tab is active, displaying a table with the following data:

N.	Obj. Name (Short)	Obj. Name (Long)	Type	Bound, All...	Bound, Us...	Reference...	Reference...	ID
1	STR	\PROGRAM=ZJANZH_TEM...	String	5.152	3.933	5.152	3.933	
2	TC_DBG_SESSION_NAME...	\PROGRAM=SAPSSY3\D...	Internal Table	2.872	464	2.872	464	
3	TC_STRUCTURE-COLS	\PROGRAM=SAPSSY3\D...	Internal Table	2.872	1.152	2.872	1.152	
4	TC_PROGRAMS-COLS	\PROGRAM=SAPSSY3\D...	Internal Table	2.872	1.324	2.872	1.324	
5	TC_MEMORY_RANKING-C...	\PROGRAM=SAPSSY3\D...	Internal Table	2.872	1.840	2.872	1.840	
6	TC_TABLE-COLS	\PROGRAM=SAPSSY3\D...	Internal Table	2.796	444	2.796	444	
7	TC_OVERVIEW-COLS	\PROGRAM=SAPSSY3\D...	Internal Table	2.796	780	2.796	780	
8	TC_EVENTS-COLS	\PROGRAM=SAPSSY3\D...	Internal Table	2.796	948	2.796	948	
9	TC_BREAKPOINTS-COLS	\PROGRAM=SAPSSY3\D...	Internal Table	2.796	1.116	2.796	1.116	
10	TC_STACK-COLS	\PROGRAM=SAPSSY3\D...	Internal Table	2.796	1.116	2.796	1.116	
11	TC_OBJECT-COLS	\PROGRAM=SAPSSY3\D...	Internal Table	2.796	1.452	2.796	1.452	

...deleted pages from 13 to 79

- This is preview version -

Read more about the complete SAP TechED 2006 Books at

WWW.SOFTBASIC.NET