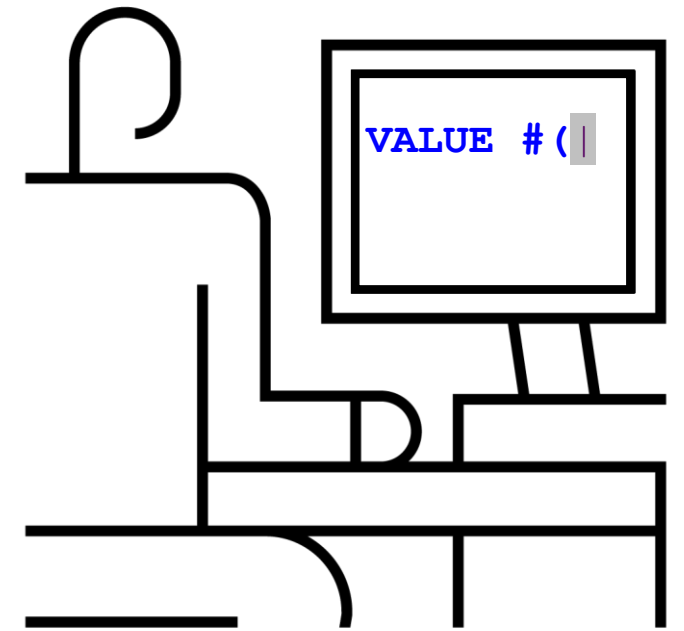


Обучение АВАР. Операторы конструкторы **VALUE** и **NEW**

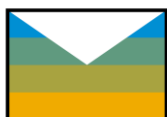
Василий Ковальский, SAP CIS
12 октября 2021 12:00 – 13:00

PUBLIC



Презентация доступна по ссылке
<https://cloud.mail.ru/public/WvfK/AgZ9Qcvwc>

Василий Ковальский,
инструктор АВАР с 1998 года



vassili.kovalski@sap.com



<https://www.facebook.com/Vassili.Kovalski>

<https://www.facebook.com/abap.education>



Серия вебинаров Обучение АВАР. Октябрь 2021



Операторы конструкторы **VALUE** и **NEW**

12 октября 2021



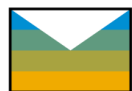
Точные вычисления. **EXACT**

13 октября 2021



Передача кластеров данных между программами. **EXPORT** и **IMPORT**

14 октября 2021



Интересны другие темы? Пишите, подумаем

Для кого этот вебинар



- в первую очередь для начинающих разработчиков
- для специалистов служб поддержки и проверки качества кода
- для всех интересующихся АВАР

О чем пойдет речь

Операторы конструкторы **VALUE** и **NEW** позволяют создавать данные любой сложности «на лету», при работе со структурами и внутренними таблицами позволяют делать очень интересные вещи. Оператор конструктор **NEW** создает объекты

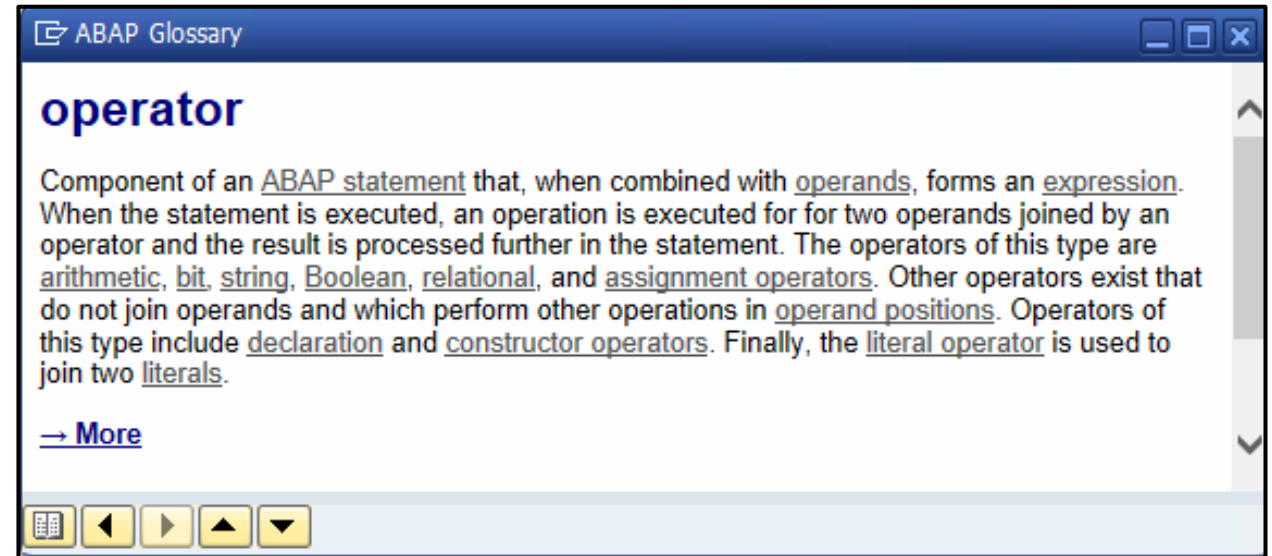
Мы обсудим:

- Основы синтаксиса
- Применение к структурам и внутренним таблицам
- Создание объектов
- Итератор **FOR**
- Где еще об этом узнать
- Что еще новое по ABAP этой осенью.



Предположительная продолжительность ~ **1 час**

“Statement” и “Operator”

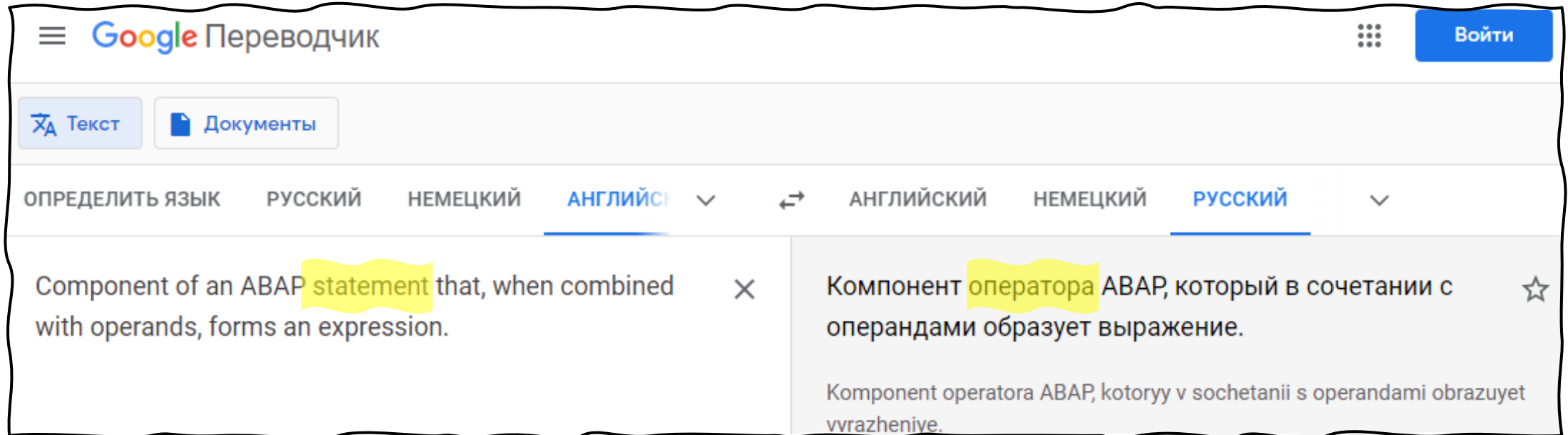


ABAP Glossary

operator

Component of an ABAP statement that, when combined with operands, forms an expression. When the statement is executed, an operation is executed for two operands joined by an operator and the result is processed further in the statement. The operators of this type are arithmetic, bit, string, Boolean, relational, and assignment operators. Other operators exist that do not join operands and which perform other operations in operand positions. Operators of this type include declaration and constructor operators. Finally, the literal operator is used to join two literals.

[→ More](#)



Google Переводчик

Текст Документы

ОПРЕДЕЛИТЬ ЯЗЫК РУССКИЙ НЕМЕЦКИЙ **АНГЛИЙСКИЙ**

АНГЛИЙСКИЙ НЕМЕЦКИЙ **РУССКИЙ**

Component of an ABAP statement that, when combined with operands, forms an expression.	Компонент оператора ABAP, который в сочетании с операндами образует выражение.
	Komponent operatora ABAP, kotoryy v sochetanii s operandami obrazuyet vyrazheniye.

Общий синтаксис операторов конструкторов

Constructor Operators for Constructor Expressions

Syntax

```
... NEW | VALUE | CONV | CORRESPONDING | CAST | REF | EXACT | REDUCE | FILTER | COND | SWITCH  
   type ( ... ) ...
```

Effect

A constructor expression consists of a

- predefined constructor operator,
- a data type or object type **type** that matches the operator and that can be derived implicitly from the operand position using #,
- and type-specified parameters specified in parentheses.

Each constructor expression creates a result whose data type is determined using the specified type. The parameters specified in parentheses are used to pass input values. The following constructor operators exist:

- The instance operator **NEW** is used to create objects in operand positions. The result is a reference variable of the static type **type** that points to the new object. The input values determine the content of the new object.
- The value operator **VALUE** is used to fill complex data objects with values in operand positions, create initial values of any data type, or control the results of table expressions. The result is a data object of the specified type **type**. The input values determine the content of the result.

Создание структуры. VALUE <<тип>>|#(...)

```
REPORT zqk_ws081_01."01. VALUE <<type>>|#( ... )
DATA
: gs_msg1 TYPE t100
, gs_msg2 TYPE t100
.
START-OF-SELECTION.
gs_msg1 = VALUE t100( sprsl = 'E' msgnr = 234 text = 'ABAP forever' ).
zcl_qk=>write_flat_structure( gs_msg1 ).
gs_msg2 = VALUE #( sprsl = 'E' arbgb = 'ZQK_WS081' text = 'I like ABAP' ).
zcl_qk=>write_flat_structure( gs_msg2 ).
SKIP.
ULINE.
WRITE: / 'BOT TAK!'.
```

```
01. VALUE <<type>>|#( ... )
```

Language	EN
Area	
Message	234
Message Text	ABAP forever

Language	EN
Area	ZQK_WS081
Message	
Message Text	I like ABAP

BOT TAK!

Создание структуры, получение ссылки. **NEW** <<тип>>|# (...)

```
REPORT zqk_ws081_02."02. NEW <<тип>>|# ( ... )
DATA
: gs_msg1 TYPE REF TO t100
, gs_msg2 TYPE REF TO t100
.
START-OF-SELECTION.
gs_msg1 = NEW t100( sprsl = 'E' msgnr = 234 text = 'ABAP forever' ).
zcl_qk=>write_flat_structure_by_ref( gs_msg1 ).
gs_msg2 = NEW #( sprsl = 'E' arbgb = 'ZQK_WS081' text = 'I like ABAP' ).
zcl_qk=>write_flat_structure_by_ref( gs_msg2 ).
SKIP.
ULINE.
WRITE: / 'BOT TAK!'.
```

01. VALUE <<тип>>|#(...)

Language	EN
Area	
Message	234
Message Text	ABAP forever

Language	EN
Area	ZQK_WS081
Message	
Message Text	I like ABAP

BOT TAK!

Создание объектов. NEW и CREATE OBJECT

```
DATA
: go_cont TYPE REF TO cl_gui_docking_container
, go_grid TYPE REF TO cl_gui_alv_grid
. . .

CREATE OBJECT go_cont
EXPORTING
  side = cl_gui_docking_container=>dock_at_bottom
  ratio = 90.

CREATE OBJECT go_grid
EXPORTING
  i_parent = go_cont.
```

Можно не создавать
ненужных переменных

```
DATA
: go_grid TYPE REF TO          cl_gui_alv_grid
. . .
go_grid = NEW cl_gui_alv_grid(
  i_parent = NEW cl_gui_docking_container(
    side = cl_gui_docking_container=>dock_at_bottom
    ratio = 90
  )
).
```

VALUE при передаче параметров

```
DATA gs_layo TYPE lvc_s_layo.  
gs_layo-zebra = 'X'.  
go_grid->set_table_for_first_display(  
  EXPORTING  
    i_structure_name = 'T100'  
    is_layout        = gs_layo  
  CHANGING  
    it_outtab       = gt  
).
```

Можно не создавать ненужных переменных

Можно присваивать значения не заранее,

а ровно в нужном месте

```
go_grid->set_table_for_first_display(  
  EXPORTING  
    i_structure_name = 'T100'  
    is_layout        = VALUE lvc_s_layo( zebra = 'X' )  
  CHANGING  
    it_outtab       = gt  
).
```

Заполнение внутренних таблиц

```
TYPES tt TYPE STANDARD TABLE OF t100 WITH EMPTY KEY.  
DATA gt TYPE tt.  
  
gt = VALUE tt(  
    ( arbgb = 'Cat' text = 'Tiglatpalassar' )  
    ( arbgb = 'Dog' text = 'Zlobermann' )  
    ( arbgb = 'Dog' text = 'Fido' )  
    ( arbgb = 'Cat' text = 'Basil' )  
).
```

06. Filling ITs

Cat	Tiglatpalassar
Dog	Zlobermann
Dog	Fido
Cat	Basil

BOT TAK!

Фраза BASE

```
DATA gt TYPE tt.  
gt = VALUE tt( ( arrgb = 'Cat' text = 'Tiglatpalassar' ) ).  
zcl_qk=>write_table( gt ).  
gt = VALUE tt( ( arrgb = 'Dog' text = 'Zlobermann' ) ).  
skip.  
zcl_qk=>write_table( gt ).  
gt = VALUE tt( BASE gt ( arrgb = 'Dog' text = 'Fido' ) ).  
skip.  
zcl_qk=>write_table( gt ).
```

07. BASE clause

Cat	Tiglatpalassar
Dog	Zlobermann
Dog	Zlobermann
Dog	Fido

BOT TAK!

Создание данных по записи внутренней таблицы

08. A record from IT

Cat	Tiglatpalassar
Dog	Zlobermann
Dog	Fido
Cat	Basil

Language	
Area	Dog
Message	
Message Text	Fido

```
TYPES tt TYPE STANDARD TABLE OF t100 WITH EMPTY KEY.
```

```
DATA: gt TYPE tt.
```

```
gt = VALUE tt(  
    ( arrgb = 'Cat' text = 'Tiglatpalassar' )  
    ( arrgb = 'Dog' text = 'Zlobermann' )  
    ( arrgb = 'Dog' text = 'Fido' )  
    ( arrgb = 'Cat' text = 'Basil' )  
).
```

```
DATA(gs) = VALUE #( gt[ text = 'Fido' ] ).
```

Создание данных по записи внутренней таблицы. Фраза **DEFAULT**

```
TYPES tt TYPE STANDARD TABLE OF t100 WITH EMPTY KEY.  
PARAMETERS p_error AS CHECKBOX.
```

```
START-OF-SELECTION.
```

```
data(gt) = VALUE tt(  
  ( argb = 'Cat' text = 'Tiglatpalassar' )  
  ( argb = 'Dog' text = 'Zlobermann' )  
  ( argb = 'Dog' text = 'Fido' )  
  ( argb = 'Cat' text = 'Basil' )  
).
```

```
IF p_error = abap_true.
```

```
  data(gs) = VALUE #( gt[ text = 'Fid@' ] ) ." ! ERROR !
```

```
ELSE.
```

```
  data(gs_default) = VALUE t100( argb = 'Horse' text = 'Sivka' ).
```

```
  gs = VALUE #( gt[ text = 'Fid@' ] DEFAULT gs_default ).
```

```
* gs = VALUE #( gt[ text = 'Fid@' ] DEFAULT VALUE #( argb = 'Horse' text = 'Sivka' ) ).
```

```
ENDIF.
```

Category	ABAP programming error
Runtime Errors	ITAB_LINE_NOT_FOUND
Except.	CX_SY_ITAB_LINE_NOT_FOUND
ABAP Program	ZQK_WS081_09

Horse
Sivka

Итератор FOR ... THEN

```
TYPES BEGIN OF ts
/   xx TYPE p DECIMALS 3
/   ff TYPE p DECIMALS 3
/   gg type string
/   END OF ts
/   tt type STANDARD TABLE OF ts WITH EMPTY KEY.

DATA(gt) = VALUE tt(
  FOR i = -10
  THEN i + 1
  WHILE i <= 10
*  UNTIL i > 10
  ( xx = i
    ff = ( exp( i / 10 ) + exp( - i / 10 ) ) / 2
    gg = `|`
      && repeat( val = ` ` occ = ( exp( i / 10 ) + exp( - i / 10 ) ) * 20 - 1 )
      && `*`
    )
  ).
```

10. FOR, THEN, WHILE, UNTIL clauses

Hyperbolic cosine: $y = \text{ch}(x)$

10,000-	1,543	x
9,000-	1,433	x
8,000-	1,337	x
7,000-	1,255	x
6,000-	1,185	x
5,000-	1,128	x
4,000-	1,081	x
3,000-	1,045	x
2,000-	1,020	x
1,000-	1,005	x
0,000	1,000	x
1,000	1,005	x
2,000	1,020	x
3,000	1,045	x
4,000	1,081	x
5,000	1,128	x
6,000	1,185	x
7,000	1,255	x
8,000	1,337	x
9,000	1,433	x
10,000	1,543	x

Итератор FOR . Селекция

```
DATA(g1) = VALUE tt(  
    FOR wa " Work area definition  
    IN gt " Source table  
    WHERE ( arbgb = 'Cat' ) " condition  
    ( wa ) " Result  
).  
  
DATA(g2) = VALUE tt(  
    FOR wa " Work area definition  
    IN gt " Source table  
    FROM 2 " From  
    TO 4 " To  
    ( wa ) " Result  
).
```

11. FOR. Selection	
Cat	Tiglatpalassar
Dog	Zlobermann
Cow	Booryonka
Dog	Fido
Cat	Basil
Selection by field values. Only Cats	
Cat	Tiglatpalassar
Cat	Basil
Selection by index. From 2 to 4	
Dog	Zlobermann
Cow	Booryonka
Dog	Fido
BOT Tak!	

Итератор FOR . Проекция

```
TYPES: tt TYPE STANDARD TABLE OF t100 WITH EMPTY KEY
      , BEGIN OF ts
      , species TYPE t100-arbgb
      , nick     TYPE t100-text
      , END OF ts
      , td TYPE STANDARD TABLE OF ts WITH EMPTY KEY
      .

DATA(g1) = VALUE td(
  FOR wa           " Work area definition
  IN gt           " Source table
  ( species = wa-arbgb
    nick    = wa-text
  )
  ) .
```

12. FOR. Projection	
Cat	Tiglatpalassar
Dog	Zlobermann
Cow	Booryonka
Dog	Fido
Cat	Basil

Projection. Only the fields you need

Cat	Tiglatpalassar
Dog	Zlobermann
Cow	Booryonka
Dog	Fido
Cat	Basil

BOT TAK!

Итератор FOR . Селекция, проекция, вычисления

```
TYPES: tt TYPE STANDARD TABLE OF t100 WITH EMPTY KEY
      , BEGIN OF ts
      , no      TYPE p DECIMALS 4
      , species TYPE t100-arbgb
      , nick    TYPE t100-text
      , END OF ts
      , td TYPE STANDARD TABLE OF ts WITH EMPTY KEY
      .

DATA(g1) = VALUE td(
  FOR wa           " Work area definition
  IN gt           " Source table
  INDEX INTO ind  " Index value definition
  WHERE ( arbgb CP 'C*' ) " Condition
  ( no      = sin( ind ) * 11
    species = wa-arbgb
    nick    = wa-text && ` is a pet of mine `
  )
  )
).
```

13. FOR. Selection, projection, calculation

Cat	Tiglatpalassar
Dog	Zlobermann
Cow	Booryonka
Dog	Fido
Cat	Basil

Selection, projection, calculation

9,2562	Cat	Tiglatpalassar is a pet of mine
1,5523	Cow	Booryonka is a pet of mine
10,5482	Cat	Basil is a pet of mine

BOT TAK!

Где еще об этом говорится: Документация

Транзакция **ABAPDOCU**

The screenshot shows the SAP ABAP Keyword Documentation interface. The search bar contains the word 'VALUE'. The left-hand navigation pane is expanded to 'ABAP - Reference' > 'Creating Objects and Values', with 'VALUE - Value Operator' highlighted in a green box. The main content area is titled 'Creating Objects and Values' and contains the following text:

The values of dynamically or statically declared data objects can be constructed using the following [constructor expressions](#):

- When anonymous data objects are created dynamically using the instance operator [NEW](#), values for all data types, particularly structured and tabular types, can be constructed and assigned to the new data object.
- The [value operator VALUE](#) can also be used to construct the content of complex data objects (structures, internal tables). This is more than can be done using the [VALUE](#) addition.

Note

Like any constructor expression, the value operator [VALUE](#) can be used in [general expression positions](#) and [functional operand positions](#), in particular the right side of an assignment to an [inline declaration](#).

Example

Inline declarations of a reference variable `dref` and a structured variable `dobj`. The instance operator `NEW` creates an anonymous data object, referenced by `dref`. Using functional method calls, the instance operator `NEW` and the value operator `VALUE` are used to construct and assign values for the components of the structures.

```
DATA(dref) = NEW struct( col1 = meth->get_col1( )
                        col2 = meth->get_col1( )

DATA(dobj) = VALUE struct( col1 = meth->get_col1(
                          col2 = meth->get_col1(
```

The SAP logo and 'SAP NetWeaver™ ABAP Documentation' are visible at the bottom of the window.

Семинар **D75AW**. Дельта-курс: с ABAP Workbench SAP NetWeaver 7.0 на SAP NetWeaver 7.5

Настройка
производительности
ABAP-программ

D75AW 3 дн.



- Разработчики
- Консультанты в области разработки

[Официальное описание](#)
[Дополнительное описание](#)



- Введение
- Окружение
- Выражения в ABAP
- Возможности внутренних таблиц
- Изменения в Open SQL
- Core Data Services (CDS)
- Каналы сообщений ABAP
- Новые инструменты анализа



Требования

Обязательно

Опыт процедурно- и объектно-ориентированной разработки на ABAP в SAP NetWeaver 7.0

<https://www.facebook.com/abap.education>



Обучение ABAP
11 июня · 🌐

Василий Ковальский . - Адаптация пользовательского ABAP-кода для перехода на S/4HANA (S4D440)
SAPLand
<https://sapland.ru/.../adaptatsiya-polizovateliskogo-abap...>

List of Checks	
General Checks	
Cloud Readiness	
Performance Checks	
Security Checks	
Syntax Check/Generation	
Robust Programming	
Programming Conventions	
S/4HANA Readiness	
• S/4HANA: Field length extensions	👉 👉
• S/4HANA: Search for database operations	👉 👉 🚩
• S/4HANA: Search for usages of simplified objects	👉 👉
• S/4HANA: Search for ABAP Dictionary enhancements	👉 👉
• S/4HANA: Search for base tables of ABAP Dictionary views	👉 👉
• S/4HANA: Search for S/4 related syntax errors	👉 👉
Metrics and Statistics	
Dynamic Tests	

Семинар **WRGCTS**. Git-enabled Change and Transport System + Piper Library. 2-3 декабря 2021

Содержание

Обзор gCTS

Базовая конфигурация gCTS

Конфигурация репозитория + упражнение

Разработка с gCTS + упражнение

Обзор инструмента непрерывной интеграции Jenkins

Обзор библиотеки Piper

Конфигурация конвейера + упражнение

gCTS и Change Request Management + упражнение

<https://training.sap.com/course/wrgcts--ru-ru>



ABAP для консультантов. Основы
Функциональная спецификация на разработку SAP-программ для новичков
Read ABAP: чтение ABAP-кода
ABAP. ООП простыми словами
Использование объектно-ориентированного программирования в ABAP
Экспресс-курс по основам быстрого формирования спецификаций на разработку
Web ABAP: OData-сервисы и их использование в UI5-приложениях
Чистый код. Как писать код, который приятно читать и легко поддерживать
Современная отладка в ABAP: через тернии к звёздам
Unit-тесты на ABAP
Multi Page-разработка и нативный биндинг данных
Шаблоны проектирования в SAP
Разработка HCM в ODATA
Оптимизация ABAP-приложений

<https://sapland.ru/click.php?id=260>

Приобретайте знания вместе с SAP Training and Adoption



Страница SAP Training and Adoption
<https://www.sap.com/cis/training-certification.html>



Актуальные новости
<https://training.sap.com/content/CIS-RUNews>



Электронный магазин семинаров
www.training.sap.com



[@SAPtvCIS](#)



[@SAPEducation](#)

Москва, Учебный центр SAP
+7 (495) 797 27 20
education.russia@sap.com