



Internship

Talent Program

APRIL – JUNE 2025

ÍNDICE

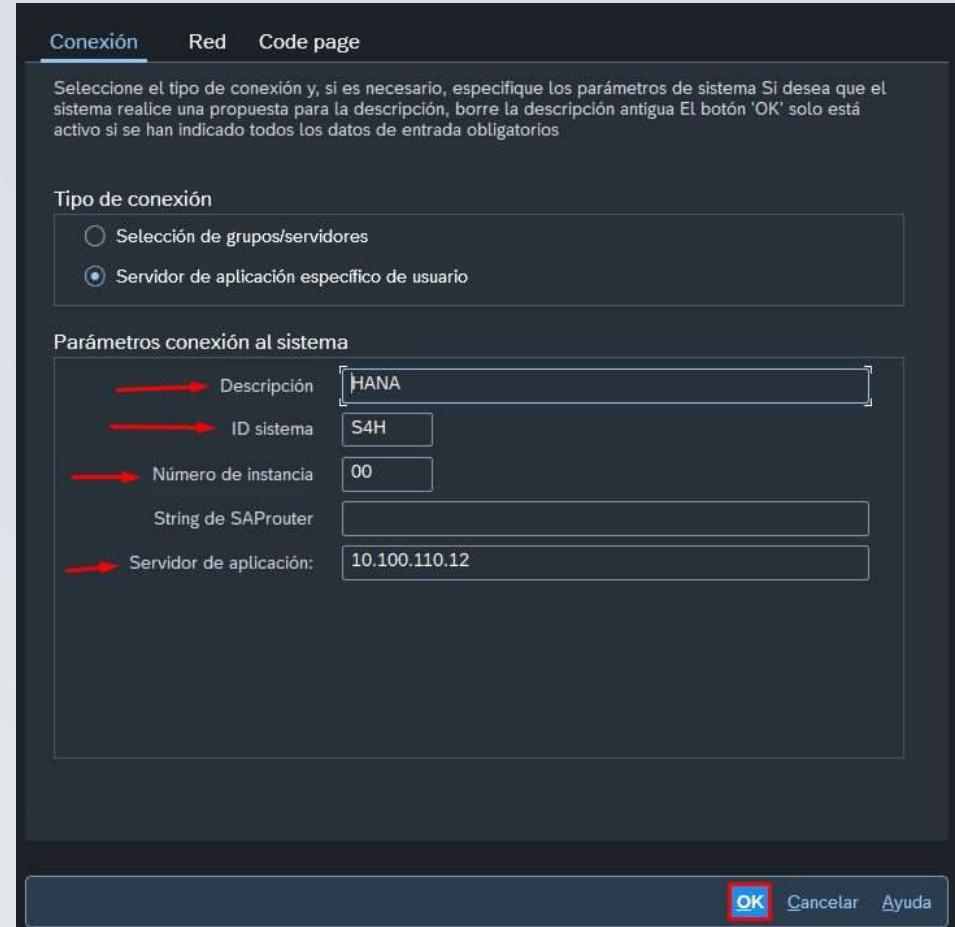
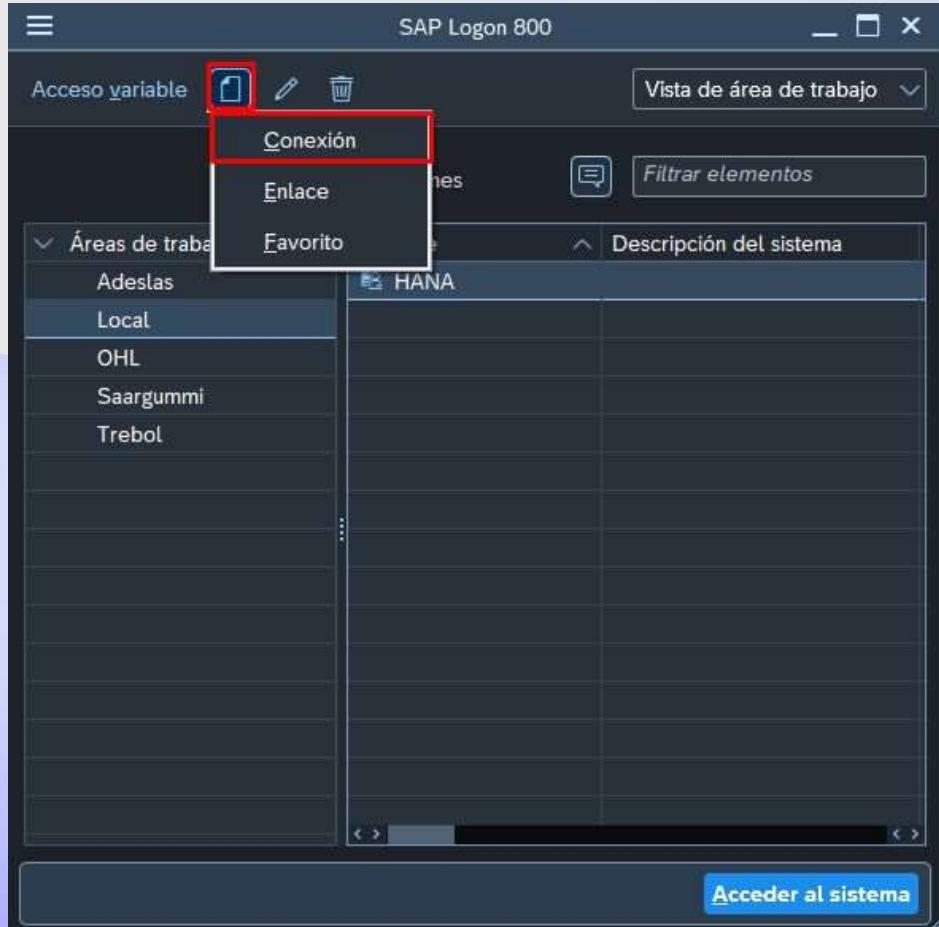
SAP Y SAP GUI
ABAP BÁSICO
BASE DE DATOS
ALV
SMART FORMS
CAP Y RAP
FIORI

¿Qué es SAP?



SAP (Systems, Applications, and Products in Data Processing) es un software empresarial que integra y gestiona procesos de negocio en una organización.

Conexión



SE38: ABAP EDITOR
SE37: FUNCTION BUILDER
SE80: OBJECT
NAVIGATOR
SE11: ABAP DICTIONARY
SE16: DATA BROWSER
SE16N: NEW DATA
BROWSER

Transacciones

EJEMPLOS:
Z3ENRAYA
ZSHOP

Crear pedido

Orden: Orden de workbench

Descripción breve: →

Proyecto:

Titular:

Status:

Última modificación: 03/12/2025 16:07:34

Mandante fuente:

Destino:

Tareas

Empleados
J.TEST

Consultar orden Workbench local

Paquete:

Orden: Orden de workbench

Descripción breve:

Órdenes propias →

H ⊖ X

TOOLS



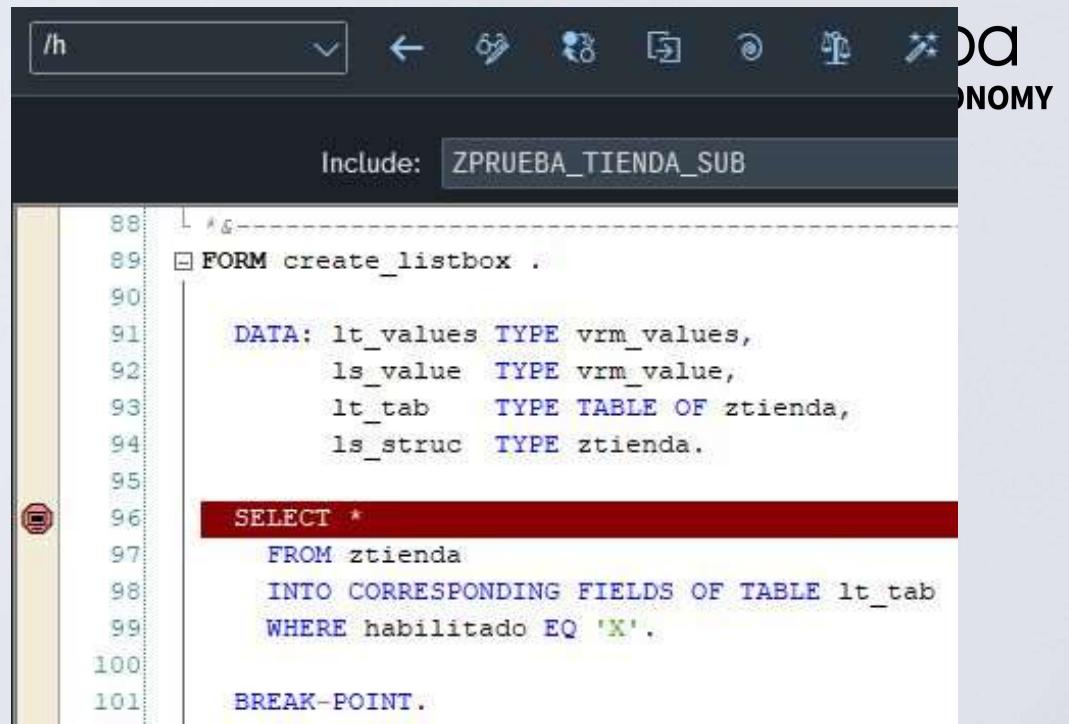
TOOLS

Debugger

ENTER

CLICK

SENTENCI
A



The screenshot shows the SAP ABAP debugger interface. The title bar includes the path '/h' and various icons. The 'Include' dropdown is set to 'ZPRUEBA_TIENDA_SUB'. The code editor displays the following ABAP code:

```
88 1 *-----  
89 2 FORM create_listbox .  
90 3  
91 4 DATA: lt_values TYPE vrm_values,  
92 5 ls_value  TYPE vrm_value,  
93 6 lt_tab    TYPE TABLE OF ztienda,  
94 7 ls_struct  TYPE ztienda.  
95 8  
96 9 SELECT *  
97 10 FROM ztienda  
98 11 INTO CORRESPONDING FIELDS OF TABLE lt_tab  
99 12 WHERE habilitado EQ 'X'.  
100 13  
101 14 BREAK-POINT.
```

A red horizontal bar highlights the 'SELECT *' statement. A small red circle with a question mark icon is positioned near the start of the code area.

ABAP

Advanced
Business
Application
Programming

Acceso directo
a base de datos

SAP GUI
ADT(Eclipse)



Comentarios y Write

```
20  * Comentario a principio de linea
21      "Comentario
22      "En
23          "Cualquier
24      "Lugar
```

```
19  DATA lv_text TYPE string VALUE 'Texto 3'.
20
21  WRITE 'Texto',
22  WRITE: 'Texto' & '2',
23  WRITE: 'Texto', '2'.
24  WRITE: lv_text.
```

Declaración de variables y sus tipos

```

5  DATA lv_var1  TYPE i.          "entero 4 bytes
6  DATA:lv_var2  TYPE string,     "cadena de caracteres dinamica
7    lv_var3  TYPE D,            "fecha formato AAAAMMDD
8    lv_var4  TYPE T,            "hora formato HHMMSS
9    lv_var5  TYPE F,            "coma flotante 8 bytes
10   lv_var6  TYPE DECFLOAT16,   "coma flotante 8 bytes
11   lv_var7  TYPE DECFLOAT34,   "coma flotante 16 bytes
12   lv_var8  TYPE XSTRING,     "secuencia de bytes de longitud dinamica
13   lv_var9  TYPE C,            "cadena de caracteres
14   lv_var10 TYPE N,           "cadena de caracteres numérica
15   lv_var11 TYPE X,           "secuencia de bytes
16   lv_var12 TYPE P.          "numero al que se le tiene que fijar la cantidad de decimales

```

```

4  □ TYPES: BEGIN OF gty_s_tipol,
5    campol      TYPE c,
6    campo2      TYPE n LENGTH 5,
7    END OF gty_s_tipol,
8    gty_t_tipol TYPE TABLE OF gty_s_tipol.

```

```
lv_suma = lv_num1 + lv_num2.  
lv_suma = 1 + 4.
```

Suma

```
lv_resto = lv_num1 MOD lv_num2.  
lv_resto = 60 MOD 7.
```

Resto



Operadores

```
lv_concatenacion = cadenal && cadena2.  
lv_concatenacion = 'cadena' && 'concatenada'.  
CONCATENATE lv_concatenacion 'mas texto' '' INTO lv_concatenacion.
```

Concatenacion

```
lv_multiplicacion = lv_num1 * lv_num2.  
lv_multiplicacion = 2 * 4.
```

Multiplicación

```
lv_division = lv_num1 / lv_num2.  
lv_division = 10 / 5.
```

Division

```
lv_resta = lv_num1 - lv_num2.  
lv_resta = 3 - 1.
```

Resta

```
lv_division = lv_num1 DIV lv_num2.  
lv_division = 10 DIV 3.
```

Div entera

```
lv_exponentacion = lv_num1 ** lv_num2.  
lv_exponentacion = 2 ** 5.
```

Exponentar

```
lv_substring = cadenal+2(3).
```

Substring



Ejercicios

Los ejercicios tienen que llamarse
`ZEJERCICIO_NUM_XX`

Ejercicio 1

- A. Declara 2 variables de cada uno de los tipos de variables de tipo numérico que has aprendido hoy y asigna les valores arbitrarios.
- B. Realiza las siguientes operaciones e imprime los resultados por pantalla entre dos variables de cada tipo que has creado: suma , resta, multiplicación, división, división entera, resto de la división y exponentación.

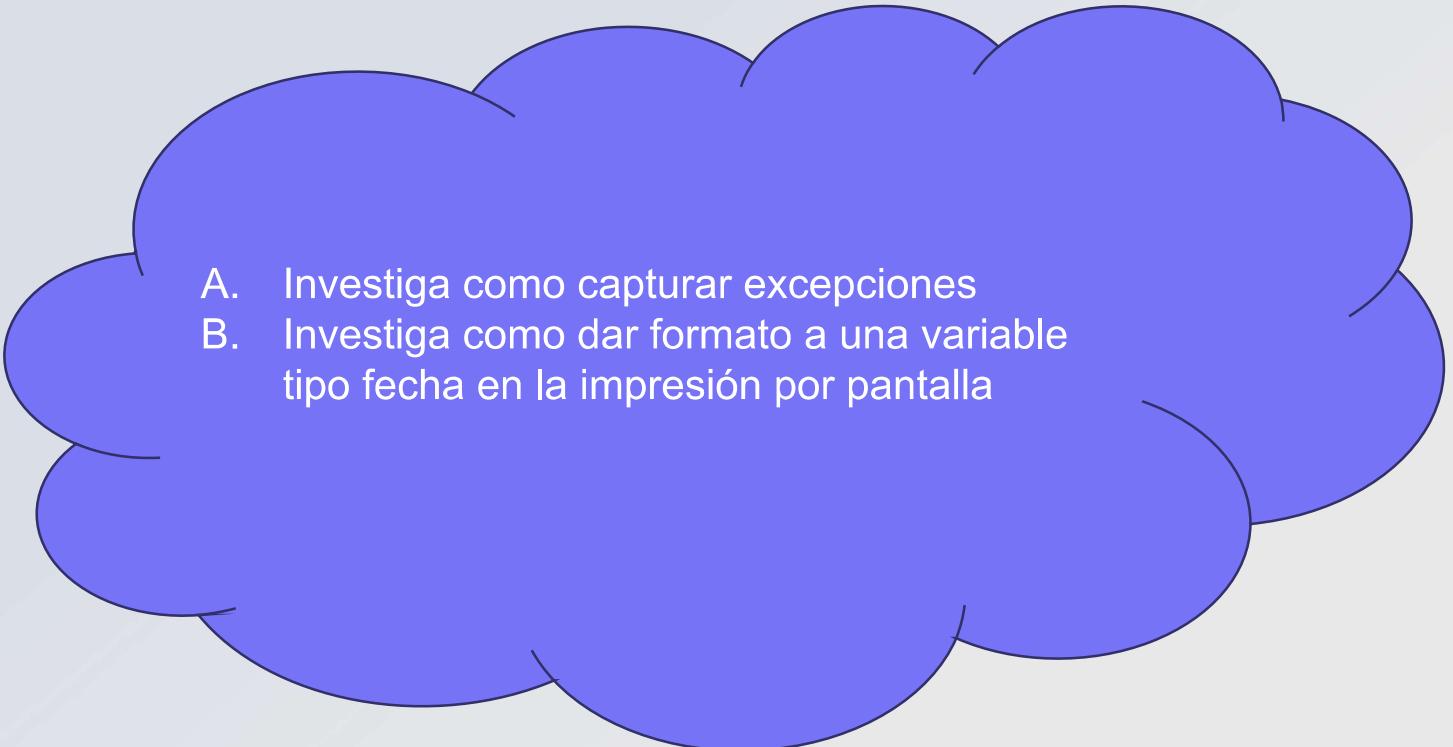
Ejercicio 2

- A. Declara 2 variables de tipo String y asigna les valores arbitrarios.
- B. Imprime por pantalla la concatenación de ambas variables
- C. Imprime por pantalla el substring de ambas variables (posiciones a elección para el substring).
- D. Imprime la concatenación de un substring de cada una de las variables (posiciones a elección para el substring).

Ejercicio 3

- A. Declara un tipo producto con los campos ID(C5), Name(C15) y UPrice(P decimals 2).
- B. Declara un tipo pedido con los campos ID(C10), Date(D), Time(T), 5X ProductX(tipo producto declarado anteriormente) y TotalPrice(P decimals 2).
- C. Ahora crea 5 variables del tipo producto creado en el apartado A y rellena una variable del tipo pedido declarado en el apartado B, teniendo en cuenta que solo hay una unidad de cada producto.
- D. Imprime por pantalla la variable de tipo pedido que has creado.

Ejercicio 4

- 
- A large, light-blue cloud-shaped callout box is positioned in the center-right area of the slide. It contains two items labeled A and B, which are listed below.
- A. Investiga como capturar excepciones
 - B. Investiga como dar formato a una variable tipo fecha en la impresión por pantalla

Basic Screen

```
4 PARAMETERS  p_num  TYPE i.  
5 PARAMETERS: p_text  TYPE string,  
6           p_check TYPE c AS CHECKBOX,  
7           p_rb0   RADIobutton GROUP rb USER-COMMAND usr,  
8           p_rbl   RADIobutton GROUP rb,  
9           p_pkm  VISIBLE LENGTH 20 as LISTBOX LENGTH 20.  
10  
11 SELECT-OPTIONS s_pkm FOR zpokemons-name.  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51 CALL FUNCTION 'VRM_SET_VALUES'  
52   EXPORTING  
53     id      = 'P_PKM_1'  
54     values  = lt_values.
```

Basic Screen

Ayuda p.búsq.elemento: ZPOKEMONS activo

Descripción breve: Ayuda de búsqueda para la tabla ZPOKEMONS

Atributos Definición

Obtención de datos

Método selección: ZPOKEMONS

Comportamiento diálogo

Tipo diálogo: Visualización de valores inmediata

Tabla de texto:

Tecla sensible:

Opciones ampliadas

Búsqueda de propuestas en campos de entrada

Búsqueda en texto completo en todas las columnas (depend.base datos)

Valor precisión p.búsq.en texto compl.toler.error: 0,8

Exit Ayuda p.búsq.:

Parámetro

Parám.Ayuda búsq.	IMP	EXP	P.I.	PosS	V.s.	Elem.datos	M...	Valor estandar
<input type="checkbox"/> ID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	1	<input type="checkbox"/>	ZPKM_ID		
<input type="checkbox"/> NAME	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	2	2	<input type="checkbox"/>	ZPKM_NAME		

aspas
aspasNETCONOMY

Tabla base datos

Vista

Tipo de datos

Grupo tipos

Dominio

Ayuda p.búsqueda ZPOKEMONS

Objeto de bloqueo

Visualizar Modificar **Crear**

TABLES: zpokemons.
SELECT-OPTIONS s_pk FOR zpokemons-name MATCHCODE OBJECT zpokemons.

Ejercicio 3.5

1. Copia el ejercicio numero 3 en otro programa y realiza los siguientes cambios:
 - Rellena la informacion de los productos con parameters y select options (usa almenos un listbox)
 - Añade a la estructura de producto un campo cantidad el cual se debe llenar con parameters tambien

Conditions

```
18 CHECK sy-subrc NE 0.  
19 WRITE:'Error'.
```

```
18 CASE sy-subrc.  
19   WHEN 1.  
20     WRITE: '1'.  
21   WHEN 2.  
22     WRITE: '2'.  
23   WHEN 3.  
24     WRITE: '3'.  
25   WHEN 4.  
26     WRITE: '4'.  
27   WHEN 5.  
28     WRITE: '5'.  
29   WHEN OTHERS."Opcional"  
30     WRITE: 'Otro'.  
31 ENDCASE.
```

```
18 IF sy-subrc EQ 0.  
19   WRITE'Sin errores'.  
20 ELSEIF sy-subrc EQ 3.  
21   WRITE'Faltan permisos'.  
22 ELSE.  
23   WRITE'Error'.  
24 ENDIF.
```

```
18  DO 1000 TIMES.  
19    CHECK ( SY-INDEX MOD 2 ) EQ 0.  
20    WRITE: sy-index.  
21  ENDDO.
```

Do y While

```
21  WHILE lv_flag EQ ''.  
22    IF sy-index EQ 2000.  
23      lv_flag = 'X'.  
24    ELSEIF ( sy-index MOD 4 ) EQ 0.  
25      WRITE: 'Año bisiesto:', sy-index.  
26    ENDIF.  
27  ENDWHILE.
```

Continue, Check y Exit

```
20  DO.
21      CHECK ( sy-index MOD 7 ) EQ 0.
22      WRITE sy-index.
23      ADD 1 TO lv_nums.
24      CHECK lv_nums LT 100.
25      CONTINUE.
26      CHECK lv_nums EQ 100.
27      WRITE 'Cien primeros multiplos de 7 encontrados'.
28      EXIT.
29  ENDDO.
```



Estructuras, sy

```
18  WRITE sy-abcde."abecedario completo"
19  WRITE sy-uname."nombre de usuario
20  WRITE sy-datum."fecha actual
21  WRITE sy-uzeit."hora actual
22  WRITE sy-tabix."indice iteracion en loop
23  WRITE sy-index."indice iteracion que no sea de loop
24  WRITE sy-langu."idioma del usuario actual
25  WRITE sy-cprog."nombre de programa en ejecucion
26  WRITE sy-repid."report en ejecucion
```

HASHED

```
9: DATA: gt_sorted TYPE SORTED TABLE OF zmov_tienda WITH UNIQUE KEY id.
```

El sistema ordena los registros automáticamente y se puede acceder a los registros mediante index y key

```
18: READ TABLE gt_sorted INDEX 1 TRANSPORTING NO FIELDS.  
19: READ TABLE gt_sorted WITH KEY fecha = sy-datum INTO gs_struc.
```

LOOP

```
11 DATA: gt_standart TYPE STANDARD TABLE OF zmov_tienda.  
12 DATA: gt_standart2 TYPE TABLE OF zmov_tienda.
```

```
16 READ TABLE gt_standart INDEX 1 TRANSPORTING NO FIELDS.  
17 READ TABLE gt_standart WITH KEY fecha = sy-datum INTO gs_struct.
```

```
12 DELETE gt_standart WHERE fecha EQ sy-datum.  
13 REFRESH gt_standart.
```

```
15 INSERT LINES OF gt_standart INTO gt_standart2.
```

```
17 APPEND LINES OF gt_standart TO gt_standart2.
```

Field Symbol

```
22 DATA: lt_test TYPE gty_t_test WITH HEADER LINE,  
23      lv_name TYPE string.  
24 FIELD-SYMBOLS <fs> TYPE gty_s_test.  
25  
26   LOOP AT lt_test.  
27     DO 10 TIMES.  
28       lv_name = |campo{ sy-index }|.  
29       ASSIGN COMPONENT lv_name OF STRUCTURE lt_test TO <fs>.  
30     ENDDO.  
31   ENDLOOP.
```

```
7   TYPES: BEGIN OF gty_s_test,  
8     campo1  TYPE i,  
9     campo2  TYPE i,  
10    campo3  TYPE i,  
11    campo4  TYPE i,  
12    campo5  TYPE i,  
13    campo6  TYPE i,  
14    campo7  TYPE i,  
15    campo8  TYPE i,  
16    campo9  TYPE i,  
17    campo10 TYPE i,  
18  END OF gty_s_test,  
19  gty_t_test TYPE TABLE OF gty_s_test.
```

Eventos de programa

```
142 INITIALIZATION. "Es el primer evento del programa
143   p_but = 'Nuevo'. "se ejecuta nada mas ejecutarlo
144   p_REF = 'Buscar'.
145   p_excel = 'Excel'.
146   p_delete = 'Borrar'.
147   p_save = 'Guardar'.

165 AT SELECTION-SCREEN OUTPUT. "Se ejecuta cuando se construye la pantalla
166   PERFORM toggle_alv_visibility.

181 AT SELECTION-SCREEN ON p_prod."Cuando se interactua con la pantalla
182   PERFORM set_unidad.

215 AT SELECTION-SCREEN ON VALUE-REQUEST FOR p_cuant. Cantidad:  
215 AT SELECTION-SCREEN ON HELP-REQUEST FOR p_cuant."F1 sobre un campo
216   LEAVE LIST-PROCESSING.

215 END-OF-SELECTION."Cuando toda la obtencion de datos se ha completado
13 START-OF-SELECTION."Cuando se presiona 'F8(Execute)'
```

Subrutina



```
2 DATA: gv_number TYPE i VALUE 10.  
3  
4 FORM change_value USING p_number TYPE i  
   CHANGING p_number2 TYPE i.  
5   p_number2 = p_number2 + p_number.  
6   WRITE: / 'Dentro de la subrutina:', p_number2.  
7 ENDFORM.  
8  
9 START-OF-SELECTION.  
10  PERFORM change_value USING 5 CHANGING gv_number.  
11  WRITE: / 'Fuera de la subrutina:', gv_number. "Sigue siendo 10"  
12
```

```
2 DATA: gv_number TYPE i VALUE 10.  
3  
4 FORM display_global.  
5   WRITE: / 'El valor global es:', gv_number.  
6 ENDFORM.  
7  
8 START-OF-SELECTION.  
9   PERFORM display_global. "Accede a gv_number directamente"  
10
```

```
2 DATA: gv_number TYPE i VALUE 10.  
3  
4 FORM change_value USING p_number TYPE i  
   p_number2 TYPE i.  
5   p_number2 = p_number2 + p_number.  
6   WRITE: / 'Dentro de la subrutina:', p_number2.  
7 ENDFORM.  
8  
9 START-OF-SELECTION.  
10  PERFORM change_value USING 5 gv_number.  
11  WRITE: / 'Fuera de la subrutina:', gv_number. "Sigue siendo 10"  
12
```

Clases y métodos



```
35 DATA go_object TYPE REF TO lcl_student.  
36 DATA gv_name      TYPE      string.  
37 DATA gv_date      TYPE      sy-datum.  
38  
39 INITIALIZATION.  
40   CREATE OBJECT go_object.  
41   CALL METHOD go_object->set_release_data( sy-datum ).  
42   go_object->set_name( 'Joao' ).  
43   gv_date = go_object->get_release_data( ).  
44   gv_name = go_object->get_name( ).  
45   WRITE: gv_date, gv_name.  
46   go_object->set_name( 'Joao2' ).  
47   WRITE: gv_date, go_object->get_name( ).
```

```
3   □ CLASS lcl_student DEFINITION.  
4     PUBLIC SECTION.  
5       CLASS-METHODS get_name          RETURNING VALUE(e_name)           TYPE string.  
6           METHODS set_name           IMPORTING    i_name             TYPE string.  
7           METHODS set_release_data  IMPORTING    i_release_data  TYPE ersda.  
8           METHODS get_release_data  RETURNING VALUE(e_release_data) TYPE ersda.  
9  
10    PROTECTED SECTION.  
11  
12    PRIVATE SECTION.  
13      CLASS-DATA: name      TYPE string.  
14      DATA:        release_data TYPE ersda.  
15  ENDCCLASS.  
16  
17  □ CLASS lcl_student IMPLEMENTATION.  
18    □ METHOD get_name.  
19      e_name = name.  
20    ENDMETHOD.  
21  
22    □ METHOD set_name.  
23      name = i_name.  
24    ENDMETHOD.  
25  
26    □ METHOD set_release_data.  
27      release_data = i_release_data.  
28    ENDMETHOD.  
29  
30    □ METHOD get_release_data.  
31      e_release_data = release_data.  
32    ENDMETHOD.  
33  ENDCCLASS.
```

Ejercicio 5

Desarrolla un programa en ABAP que permita gestionar un inventario de productos y realizar pedidos. El programa debe permitir que el usuario agregue productos hasta que decida finalizar el pedido, para hacer esta tienes que investigar el uso de botones.

Creación de tablas



SAP SE1 interface showing the creation of a new table (ZTEST).

Table properties:

- Tabla base datos: ZTEST
- Vista: none
- Descripción breve*: Tabla de prueba
- Atributos: none
- Entrega y actualización: none
- Campos: none
- Ayuda p./Verif.entr.: none
- Campos de moneda/cantidad: none
- Índices: none

Delivery class: A Tabla aplicación (datos maestros y de movimiento)

Browser data/Actual.view table: Visual./Actual.permitted

Table structure (Elem.datos tab):

	Clv	Val.i.	Elem.datos	Tipo de datos	Long.	Decima.	Sist.coord.	Descripción breve
Parámetros de memoria lógicos	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	MANDT	CLNT	3	0		0 Mandante
	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		CHAR	5	0		0 Login ID
Clase de datos: * APPL1	ED	<input type="checkbox"/>	<input checked="" type="checkbox"/>	CHAR	10	0		0 Usuario loggeado
Categ.tamaño: [1] [Q]	ED	<input type="checkbox"/>	<input checked="" type="checkbox"/>	DATS	8	0		0 Fecha login
	ED	<input type="checkbox"/>	<input checked="" type="checkbox"/>	TIMN	6	0		0 Hora login

Actualizadores de tablas



The screenshot shows the SAP SE54 interface with the 'Utilidades' tab selected. In the 'Generador actualiz.tabla' section, the 'Generador actualiz.tabla' option is highlighted. The 'Browser de Repository' shows 'ZTEST' selected under 'Grupo de funciones'. The code editor on the right displays ABAP code for a dynpro named 0100.

```
PROCESS BEFORE OUTPUT.  
MODULE LISTE_INITIALISIEREN.  
LOOP AT EXTRACT WITH CONTROL.  
  TCTRL_ZTEST CURSOR NEXTLINE.  
  MODULE LISTE_SHOW_LISTE.  
ENDLOOP.  
  
PROCESS AFTER INPUT.  
MODULE LISTE_EXIT_COMMAND A.  
MODULE LISTE_BEFORE_LOOP.  
LOOP AT EXTRACT.  
  MODULE LISTE_INIT_WORKARE_CHAIN.  
  FIELD ZTEST-ID .  
  FIELD ZTEST-USER_LOGGED  
  FIELD ZTEST-DATE_LOGGED  
  FIELD ZTEST-TIME_LOGGED  
  MODULE SET_UPDATE_FLAG O  
ENDCHAIN.  
FIELD VIM_MARKED MODULE L  
CHAIN.
```

Sentencias OpenSQL

```
2 DATA: gt_tabla TYPE TABLE OF ZTEST WITH HEADE
3
4 SELECT * "Si no es select single tiene que se
5   FROM ZTEST
6   INTO TABLE gt_tabla."Se puede usar APPENDIN
7   "los resultados a la ta
8 SELECT SINGLE USER_LOGGED TIME_LOGGED
9   FROM ZTEST
10  INTO CORRESPONDING FIELDS OF gt_tabla"Tambien se usa APPENDING, no es single no se pone table
11  WHERE date_logged EQ sy-datum."Se puede usar =, EQ, <, LT, >, GT, <=,
12          "LE, >=, GE, <>, NE, IN, BETWEEN x AND y,
13          " LIKE, IS [NOT] NULL, AND, OR, NOT
```

```
SELECT SINGLE MAX( time_logged )"Tambien existen AVG, COUNT,
FROM ztest
INTO CORRESPONDING FIELDS OF gt_tabla.
"FIRST, LAST, MAX, MIN, SUM
```

```
SELECT *
FROM ztest
INTO CORRESPONDING FIELDS OF TABLE gt_tabla
WHERE DATE_LOGGED EQ sy-datum
ORDER BY time_logged ASCENDING.
```

```
SELECT f~carrid b~connid ...
FROM scarr AS f INNER JOIN spfli AS b
  ON f~carrid = b~carrid
WHERE ...
```

Sentencias OpenSQL



```
2   DATA: gt_tabla TYPE TABLE OF ztest WITH HEADER LINE.  
3  
4   INSERT ztest FROM gt_tabla.  
5   INSERT ztest FROM TABLE gt_tabla.  
6  
7   MODIFY ztest FROM TABLE gt_tabla.  
8   MODIFY ztest FROM gt_tabla.  
9  
10  DELETE FROM ztest WHERE date_logged = sy-datum.  
11  DELETE ztest FROM TABLE gt_tabla.
```

Ejercicio 6

Crea una tabla de base de datos de productos de una tienda (ID , precio, stock, categoría->(ID de categoría relacionado con otra tabla de base de datos)) y su actualizador de tabla.
Después crea un programa que obtenga productos filtrando con parameters por pantalla.

The screenshot shows an SAP ABAP code editor with a table preview window. The code is demonstrating how to create an ALV (Advanced List View) display.

Code:

```

DATA: gr_alv,
      gr_fu,
      gr_co,
      lt_da.

TRY.
  CALL METHOD
    IMPORTING
      r_100.
  CHANGING
    t_table      = lt_data.

  gr_func = gr_alv->get_functions().
  gr_func->set_all( abap_true ).

  CATCH cx_salv_msg.

ENDTRY.

```

ALV Preview:

Manda...	KEY1	KE...	Pais/R...	Cludad sal...	Iniciar	Pais/R...	Ciudad de llegada	Destino	Dur.vu...	Salida	Llegada	Dista...	Dist	Chart...	Día(...
100	AA	17	US	NEW YORK	JFK	US	SAN FRANCISCO	SFO	6:01	11:00:00	14:01:00	2.572	MI		0
100	AA	64	US	SAN FRANCISCO	SFO	US	NEW YORK	JFK	5:21	09:00:00	17:21:00	2.572	MI		0
100	AZ	555	IT	ROME	FCO	DE	FRANKFURT	FRA	2:05	19:00:00	21:05:00	845	MI		0
100	AZ	788	IT	ROME	FCO	JP	TOKYO	TYO	12:55	12:00:00	08:55:00	6.130	MI		1
100	AZ	789	JP	TOKYO	TYO	IT	ROME	FCO	15:40	11:45:00	19:25:00	6.130	MI		0
100	AZ	790	IT	ROME	FCO	JP	OSAKA	KIX	13:35	10:35:00	08:10:00	6.030	MI	X	1
100	DL	106	US	NEW YORK	JFK	DE	FRANKFURT	FRA	7:55	19:35:00	09:30:00	3.851	MI		1
100	DL	16...	US	NEW YORK	JFK	US	SAN FRANCISCO	SFO	6:22	17:15:00	20:37:00	2.572	MI		0
100	DL	19...	US	SAN FRANCISCO	SFO	US	NEW YORK	JFK	5:25	10:00:00	18:25:00	2.572	MI		0
100	JL	407	JP	TOKYO	NRT	DE	FRANKFURT	FRA	12:05	13:30:00	17:35:00	9.100	KM		0

ALV básico

```

31   gr_column->set_medium_text( 'KEY2' ).
32
33   gr_columns->set_optimize( abap_true ).
34
35   gr_alv->display( ).
```

Ejercicio 7

Haz que el programa creado en el ejercicio 6 muestre el resultado de la búsqueda de productos con un ALV GRID. Después copia el ejercicio 6 en otro programa (ejercicio 7) y haz que se muestre el resultado en un ALV básico.

Ejercicios Finales

Ejercicio Final 1

Crea una tabla en SAP que almacene tus Pokémon favoritos a partir de la tabla `ZPOKEMONS`. Esta tabla debe incluir:

- Número en la Pokédex
- Nombre del Pokémon
- Tipos (un Pokémon puede tener uno o dos tipos)
- Habilidades (de una a tres habilidades por Pokémon)

Ejercicio Final 2

Desarrolla un programa en SAP que permita seleccionar y guardar Pokémon favoritos en la base de datos. Funciones principales:

1. Permitir al usuario seleccionar un Pokémon de los existentes en la tabla ZPOKEMONS.
2. Guardar el Pokémon seleccionado en la tabla de favoritos creada en el ejercicio 1.
3. Mostrar los Pokémon favoritos guardados en un ALV (ABAP List Viewer).

Ejercicio Final 3

Desarrolla un programa en el que puedas:

- Crear y guardar equipos Pokémon en la base de datos
- Cada equipo debe contener exactamente 6 Pokémon, seleccionados de la tabla `ZPOKEMONS`.
- Cada Pokémon debe tener una habilidad (seleccionada de la tabla `ZHABILIDADES` entre sus opciones disponibles).
- Opcionalmente, cada Pokémon puede llevar un ítem (seleccionado de la tabla `ZITEMS`).
- Visualizar los equipos en un ALV, mostrando su nombre y los detalles de sus integrantes.

Tabla transparente: activo

Descripción breve: Tabla con datos para los alumnos de prácticas

Tabla transparente: activo

Descripción breve: Tabla con datos para los alumnos de prácticas

Campos	Entrega y actualización			Ayuda p./Verif.entr.			Campos	Entrega y actualización			Ayuda p./Verif.entr.			Campos de moneda/cantit.
   	   		Ay.búsq.	   	   		Ay.búsq.		Tipo instalado					
Campo	Clv	Val.i...	Elem.datos	Tipo de datos	Long.	Campo	Clv	Val.i...	Elem.datos	Tipo de datos	Long.	Decima...	Sist.c...	
ID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		CHAR	5	ID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		CHAR	5	0		
NAME	<input type="checkbox"/>	<input type="checkbox"/>		CHAR	15	NAME	<input type="checkbox"/>	<input type="checkbox"/>		CHAR	15	0		
DESCR	<input type="checkbox"/>	<input type="checkbox"/>		CHAR	150	DESCR	<input type="checkbox"/>	<input type="checkbox"/>		CHAR	150	0		

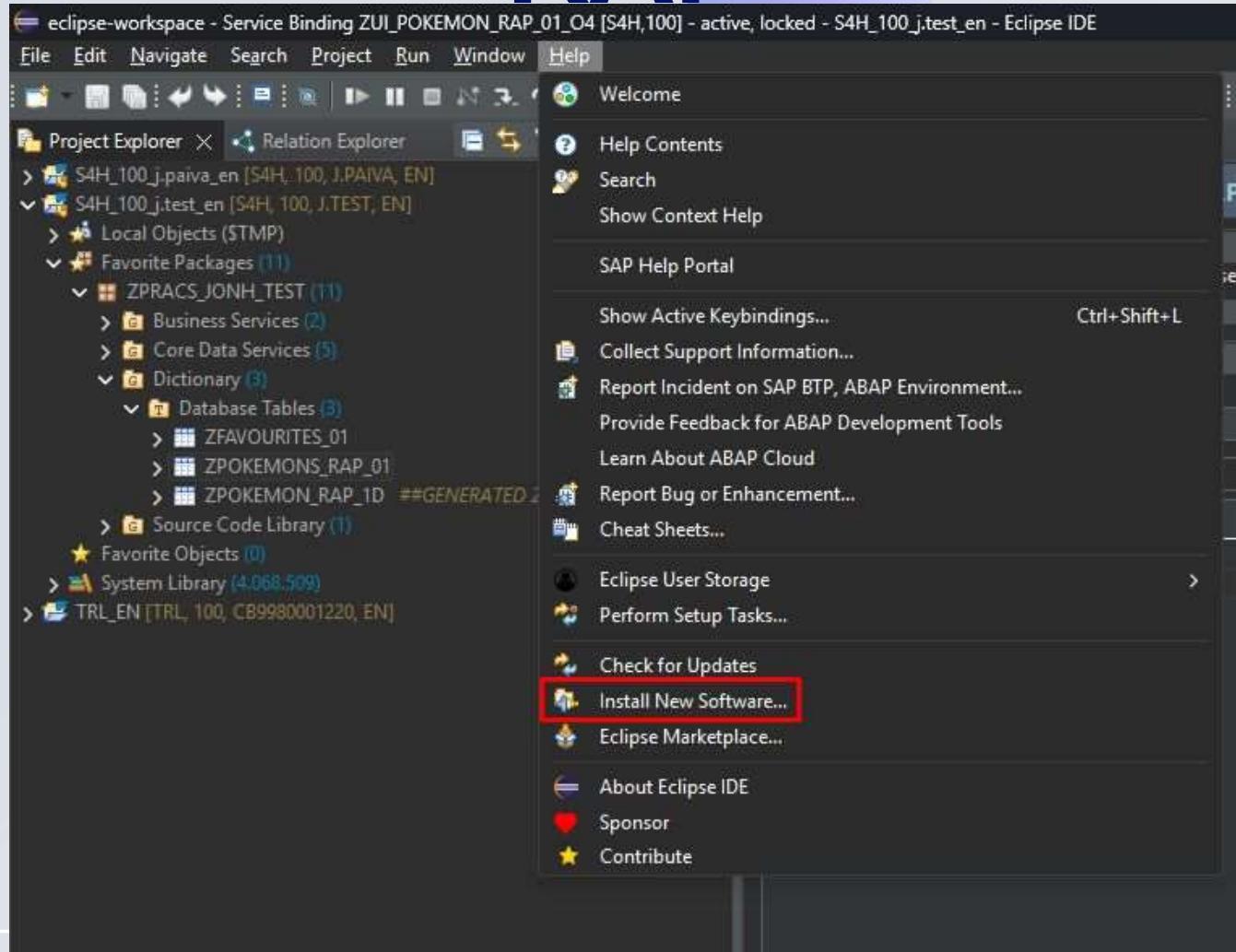
Ejercicio Final 4

Desarrolla un programa que permita organizar un torneo entre equipos Pokémon creados en el ejercicio anterior. El usuario podrá seleccionar 4, 8 o 16 equipos para competir .El informe del torneo se generará en SmartForms con la siguiente estructura:

- Hoja 1: Información general del torneo y los equipos participantes.
- Resto de hojas: Una hoja por equipo, mostrando:
 - Número en la Pokédex
 - Nombre
 - Tipos
 - Habilidad seleccionada y su descripción
 - Ítem equipado y su descripción

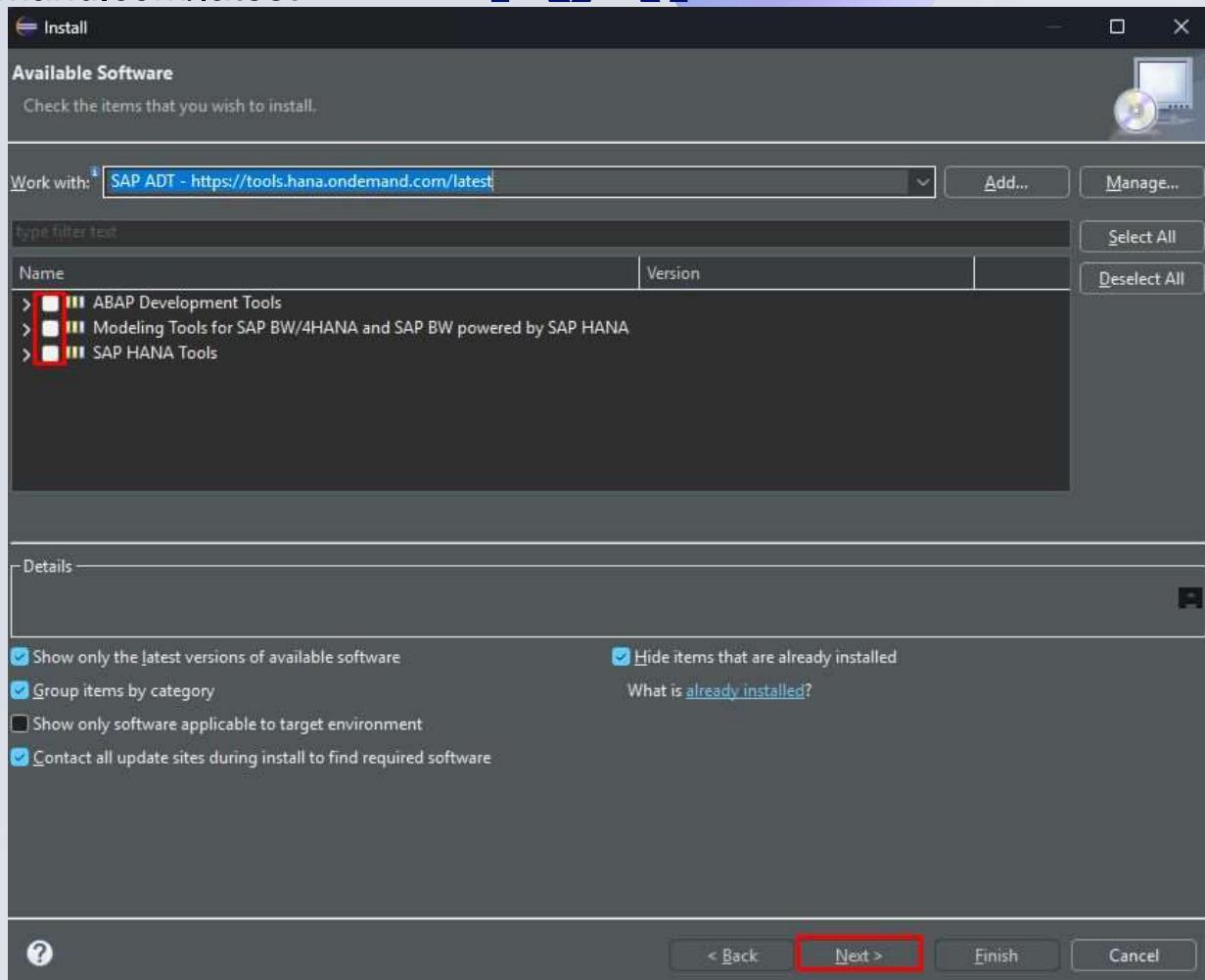
RAP CAP Y FIORI

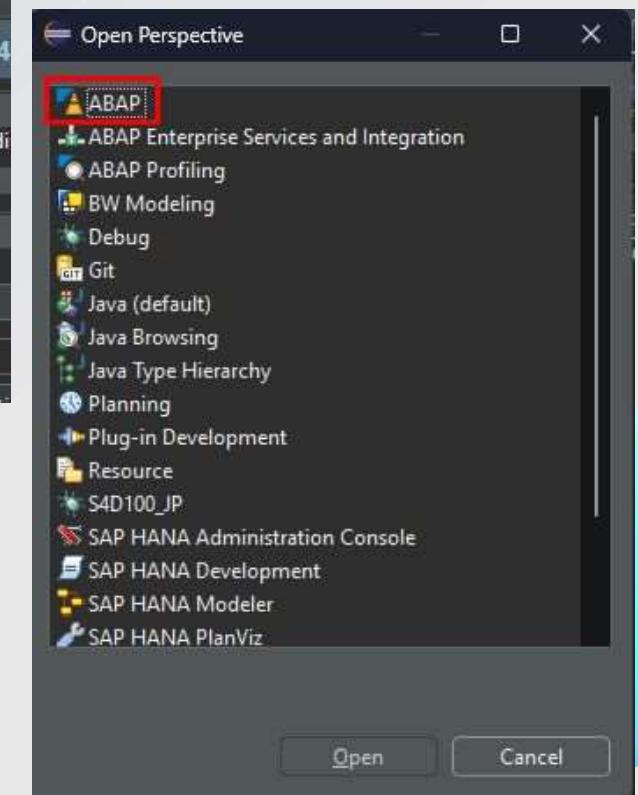
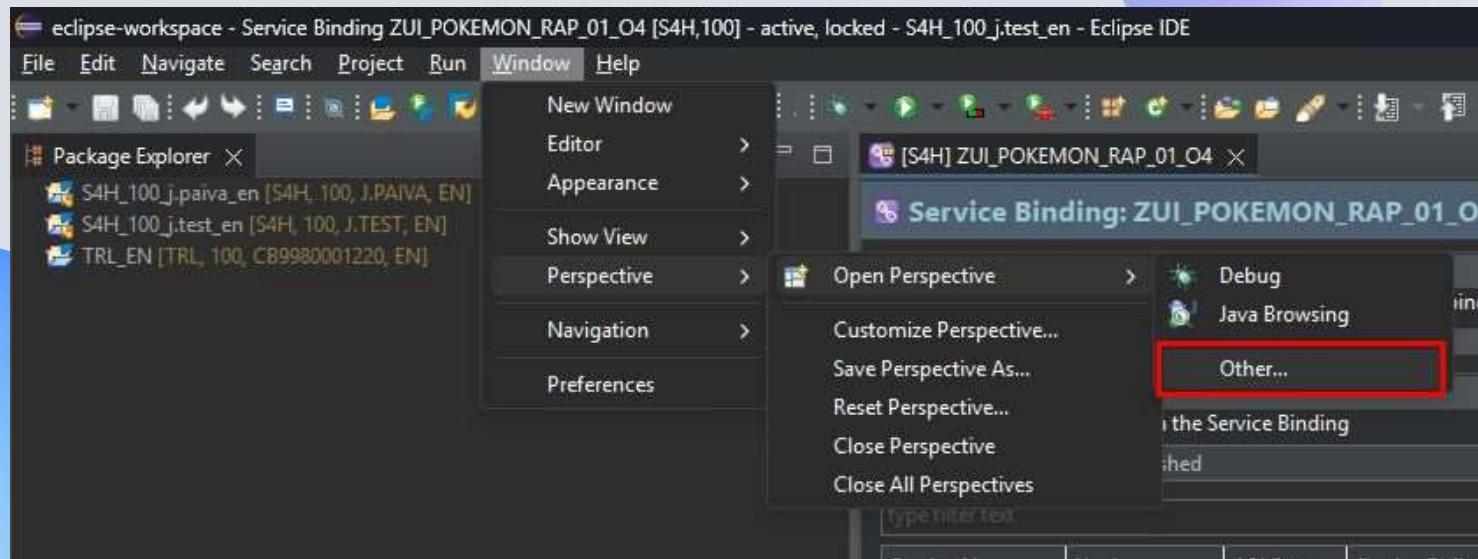
RAP



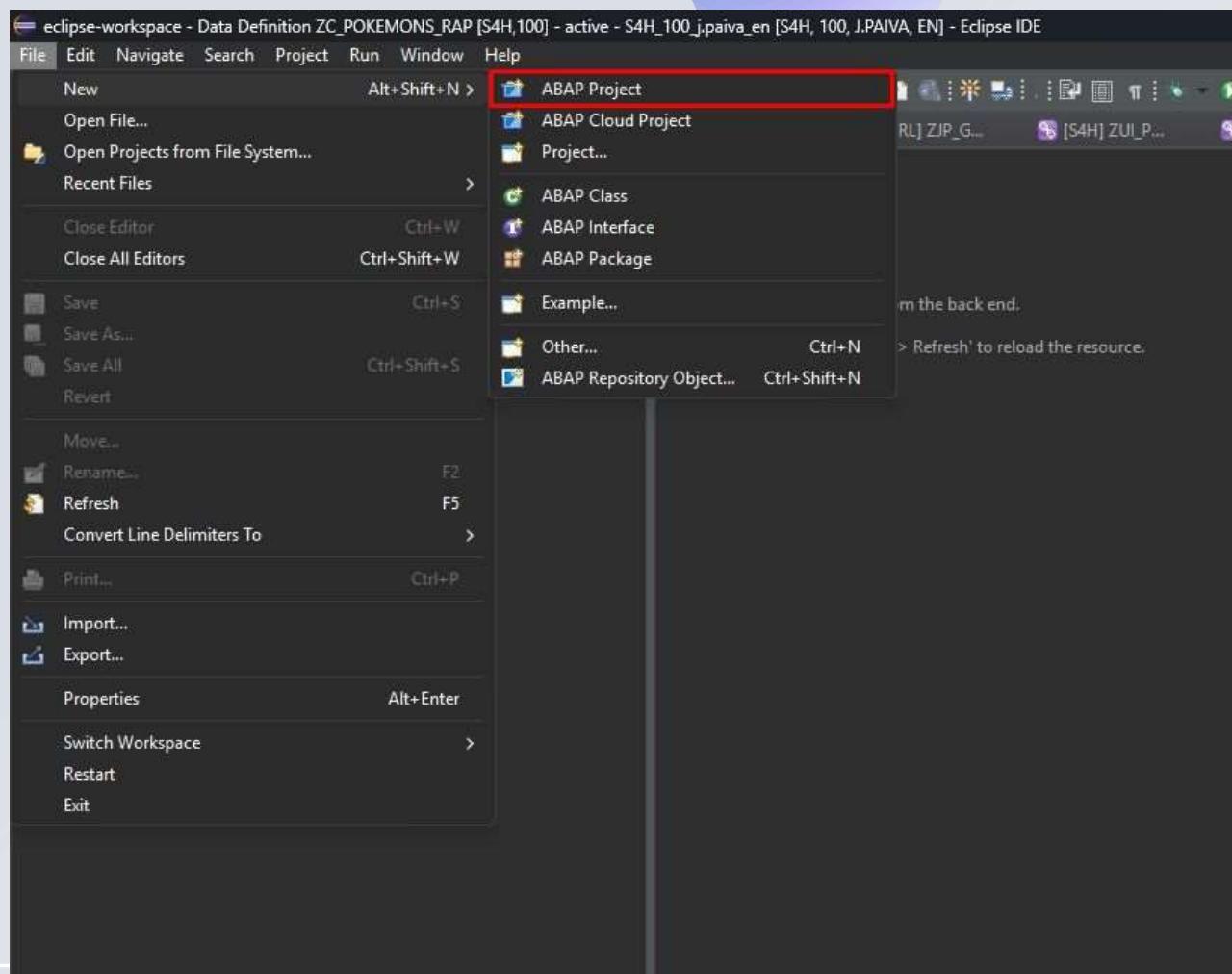
SAP ADT -
<https://tools.hana.ondemand.com/latest>

RAP





RAP



New ABAP Project

System Connection

Associate the new project with an SAP system connection

Define a [new system connection](#) from scratch, or select an existing SAP Logon entry from the list:

Name	Description	SID	Group/Server	Instance Number
HANA		S4H	10.100.110.12	00

Buttons: ? < Back Next > Finish Cancel

New ABAP Project

Connection Settings

System connection configuration for the new ABAP project

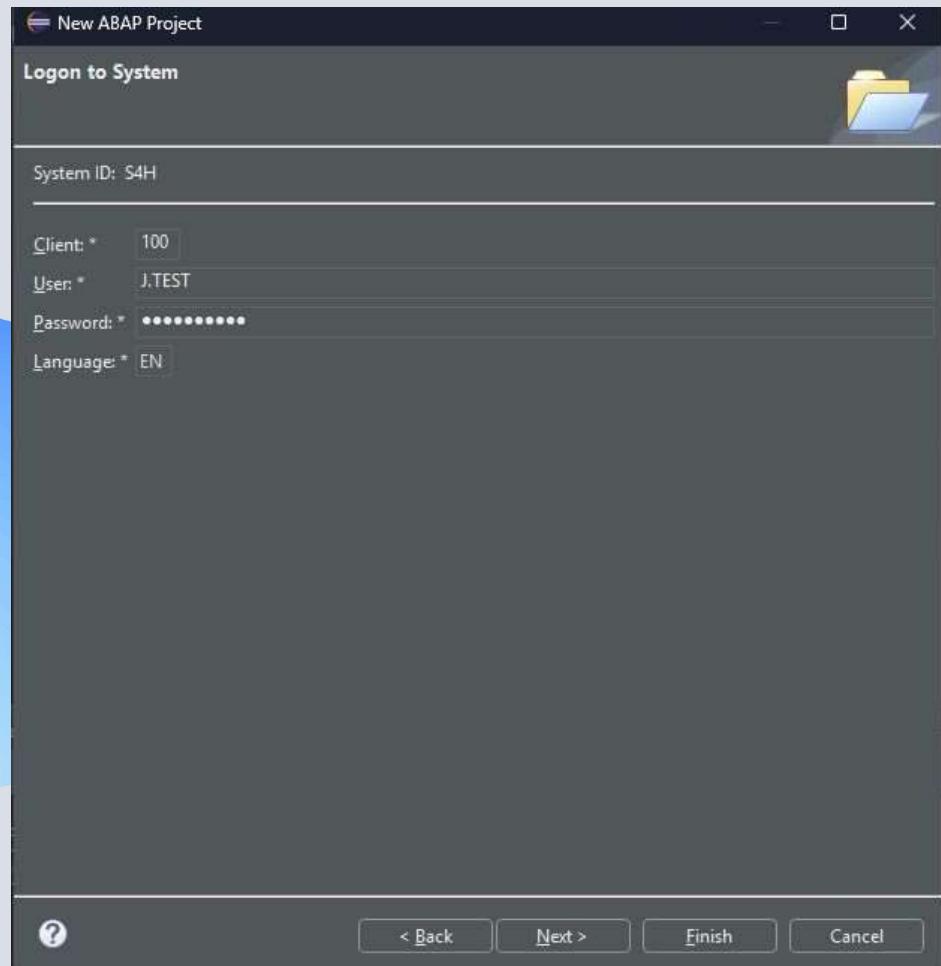
Connection Parameters

- System ID: * S4H
- Connection Type: Custom Application Server
- Message Server: *
- Group: *
- Message Server Port: *
- Application Server: * 10.100.110.12
- Instance Number: * 00
- RFC Gateway Server: *
- RFC Gateway Server Port: *
- SAProuter String: *

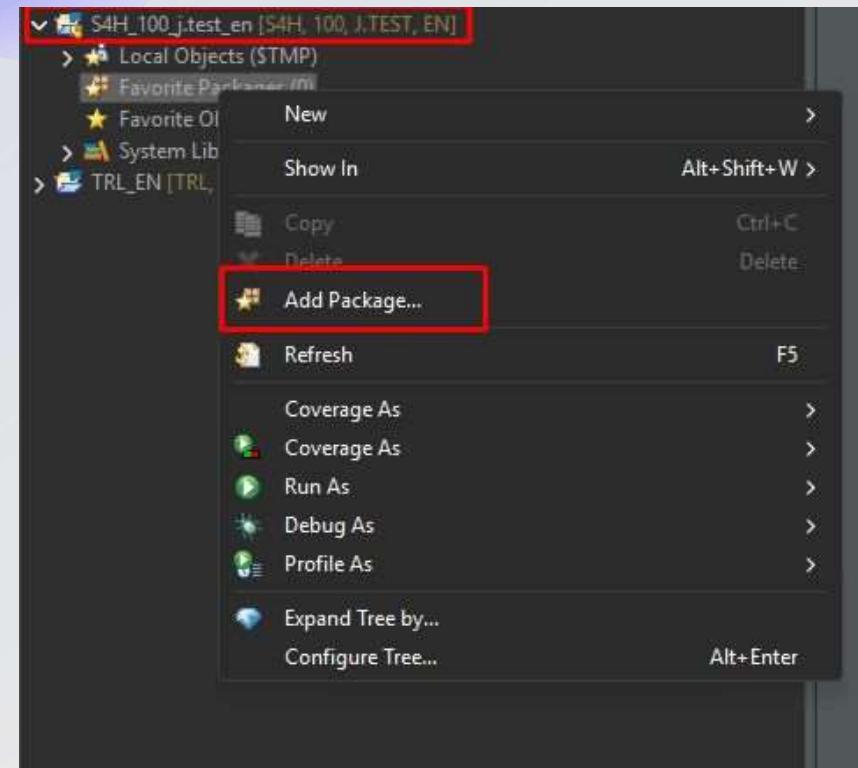
Secure Network Settings

- Activate Secure Network Communication (SNC)
- SNC Level: *
- SNC Name: *
- Single Sign-On (SSO): Disabled

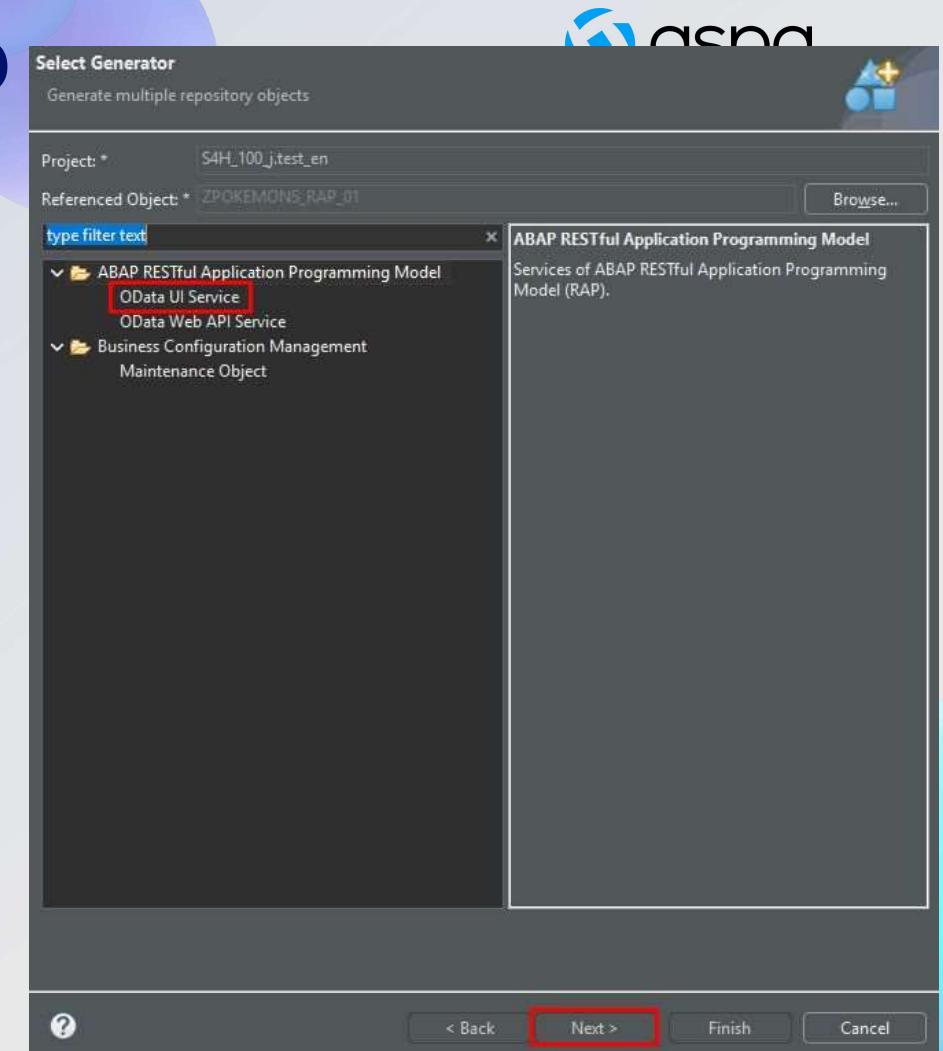
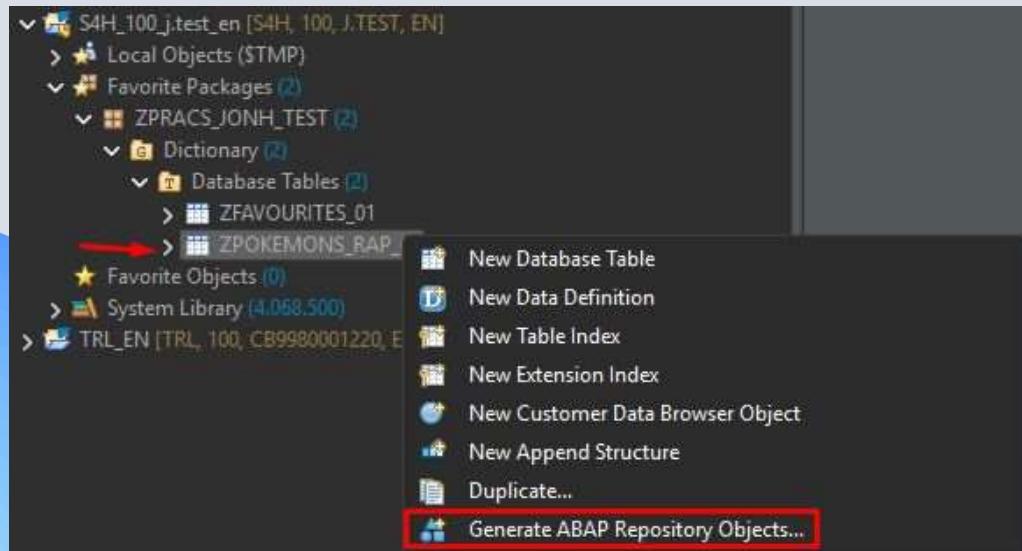
Buttons: ? < Back Next > Finish Cancel



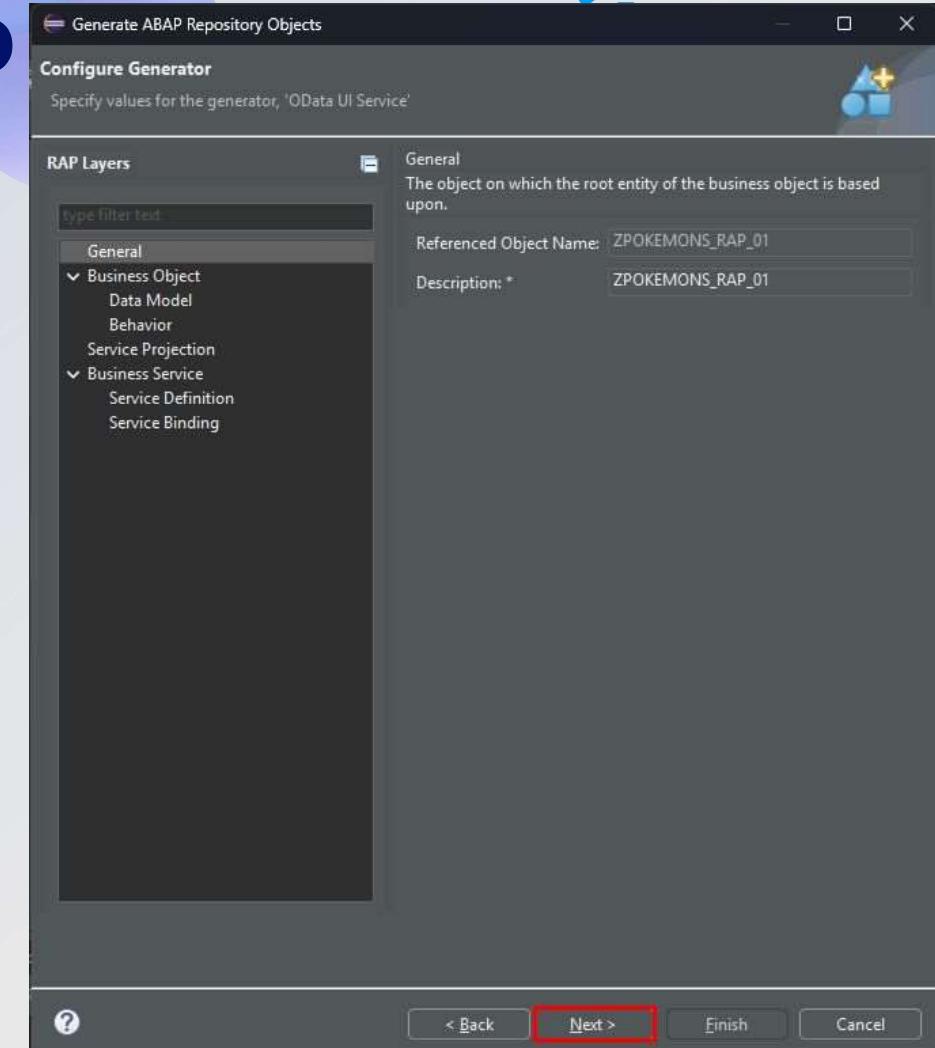
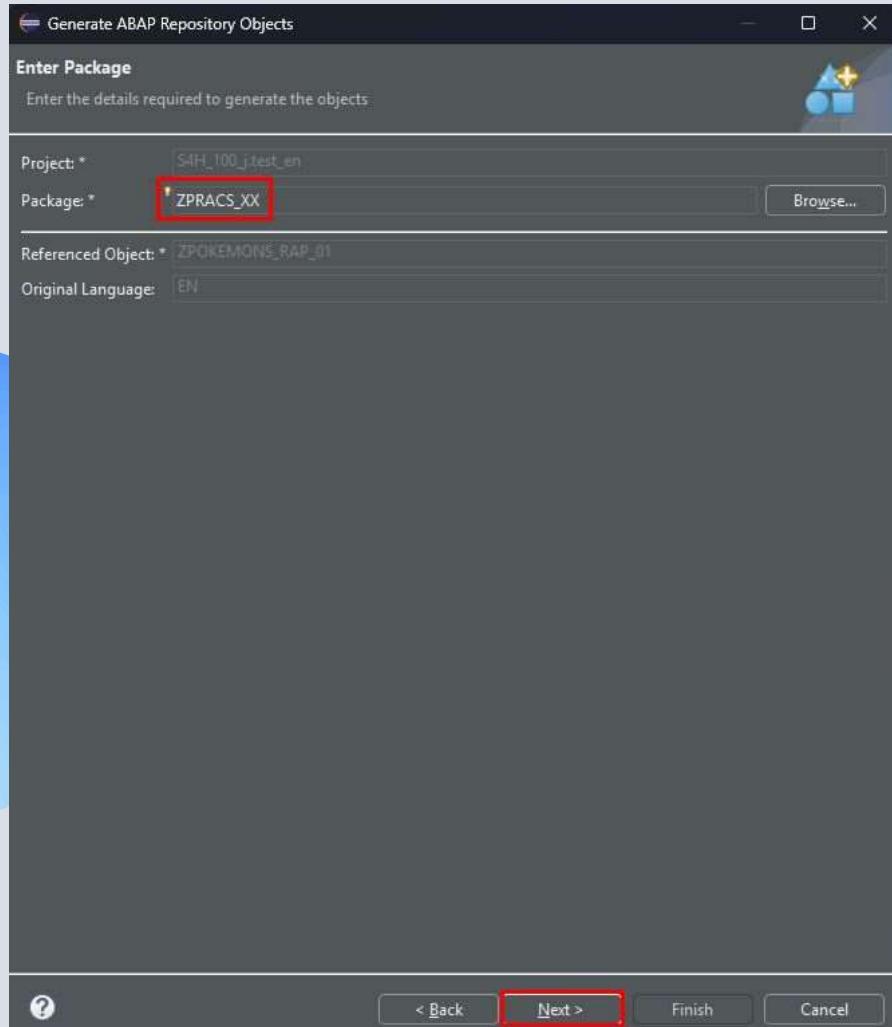
RAP



RAP



RAP



RAP



[S4H] ZUI_POKEMON_RAP_01_04 X

Service Binding: ZUI_POKEMON_RAP_01_04

General Information
This section describes general information about this service binding
Binding Type: OData V4 - UI

Services
Define services associated with the Service Binding
Local Service Endpoint: Unpublished

Type filter text:

Service Name	Version	API State	Service Definition
ZUI_POKEMON_RAP_01_04	1.0.0	Not Released	ZUI_POKEMON_RAP_01_04

Service Version Details
View information on selected service version
Service Information
Local service endpoint is not published. [Publish local service endpoint](#) to view service URL, entity sets and associations.

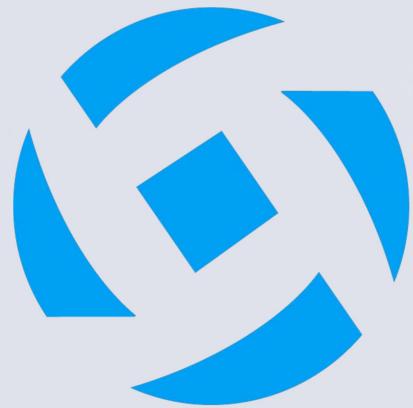
Publish (button highlighted with a red box)

Add Service... Remove

<https://10.100.110.12:44301/>



Thank you



aspaN