



Rupesh Kadge
SAP Technical Consultant



SAP BTP

WHAT IS VIRTUAL ELEMENT CONCEPT & HOW TO USE IN RAP ?



Rupesh Kadge
SAP Technical Consultant



What is Virtual elements in RAP (RESTful Application Programming) ?

A virtual element refers to the addition of a new field in an existing application without altering the current CDS entity structure. The values for this element are determined by the business logic defined in a class and interface. To implement this additional field, you must specify it using annotations.

Key Aspects of Virtual Elements

- **Non-Intrusive:** Add without altering existing CDS structures, keeping data models intact.
- **Dynamic Values:** Driven by business logic in classes and interfaces for flexibility.
- **Annotation-Based:** Use annotations for easy field definition and better readability.
- **Complexity Abstraction:** Simplify business requirements, allowing focus on high-level logic.



Rupesh Kadge
SAP Technical Consultant



- **Modularity:** Enable field additions without changing entities for easier updates.
- **Seamless Integration:** Smoothly connect with existing APIs, ensuring a consistent interface.
- **Performance:** Efficient implementation without schema changes, supporting rapid evolution.
- **Business Logic Support:** Tailor responses based on user interactions through encapsulated logic.

Step 1: You can add virtual element in required project view using below annotation where you specify the field and class name to populate value based on logic from provided class.

```
5 define view entity zc_rap_course_02 as projection on zi_rap_course_02
6 {
7   @EndUserText.label: 'Student ID'
8   key Id,
9     @EndUserText.label: 'Placed Company'
0     Placecompany,
1     @EndUserText.label: 'Project'
2     Project,
3     @EndUserText.label: 'Final Result'
4     Fresult,
5       @Semantics.systemDateTime.localInstanceLastChangedAt: true
6     Lastchangedat,
7     @ObjectModel.virtualElementCalculatedBy: 'ABAP:ZCL_ADDFIELD'
8     @EndUserText.label: 'Greetings'
9     virtual Remark : abap.char( 10 ),
0       /* Associations */
1     _student: redirected to parent zc_rap_stud_02
2 }
3 }
```



Rupesh Kadge
SAP Technical Consultant



Step 2: You need to specify the position of the additional field on your screen. Ensure that you include the field's position in the metadata of the relevant projection view where you have added the field annotation

```
33     identification: [{ position: 40, label: 'Result' }] }
34   Fresult;
35+ @UI: { lineItem: [{ position: 50, label: 'Result' }],
36   identification: [{ position: 50, label: 'Result' }] }
37   Remark;
38 }
```

Step 3: The class you added in annotation has to use interface to populate additional field value.

```
1*CLASS zcl_addfield DEFINITION
2  PUBLIC
3  FINAL
4  CREATE PUBLIC .
5
6  PUBLIC SECTION.
7    INTERFACES if_sadl_exit_calc_element_read.
8
9  PROTECTED SECTION.
10 PRIVATE SECTION.
11 ENDCLASS.
12
13
14
15*CLASS zcl_addfield IMPLEMENTATION.
16+ METHOD if_sadl_exit_calc_element_read~calculate.□
17
18+ METHOD if_sadl_exit_calc_element_read~get_calculation_info.
19
20  ENDMETHOD.
21
22 ENDCLASS.
```

Rupesh Kadge
SAP Technical Consultant



Step 4: The additional field value is based on your business logic. But make sure to add prior check if the record is not available. Since this method will trigger numerous time.

```
6* METHOD if_sadl_exit_calc_element_read~calculate.  
7  
8  DATA: lt_addfield TYPE STANDARD TABLE OF zc_rap_course_02 WITH DEFAULT KEY,  
9    lt_final     TYPE STANDARD TABLE OF zc_rap_course_02 WITH DEFAULT KEY.  
0  
1  lt_addfield = CORRESPONDING #( it_original_data ).  
2  
3  DATA(lv_mark) = REDUCE char1( INIT lv_check TYPE char1  
4    FOR <ls_mark> IN lt_addfield  
5    NEXT lv_check = COND #( WHEN <ls_mark>-Id IS NOT INITIAL  
6      AND <ls_mark>-project IS NOT INITIAL  
7      THEN abap_true  
8      ELSE abap_false ) ).  
9  
0* IF lv_mark IS NOT INITIAL.  
1  
2  lt_final = VALUE #( FOR ls_field IN lt_addfield  
3    ( Id                  = ls_field-id  
4      Placecompany       = ls_field-Placecompany  
5      Project            = ls_field-Project  
6      Lastchangedat     = ls_field-Lastchangedat  
7      Fresult            = ls_field-Fresult  
8      Remark             = COND #( WHEN ls_field-Project IS NOT INITIAL  
9        THEN 'Excellent'  
0        ELSE 'Good' ) ) ).  
1  
2  ct_calculated_data = CORRESPONDING #( lt_final ).  
3 ENDIF.
```



Rupesh Kadge
SAP Technical Consultant

Step 5: The debug prospective will give you clear idea how prior validation and additional field value is populated based on business logic.

The screenshot shows the SAP ABAP IDE interface. On the left, the code editor displays an implementation class for ZCL_ADDFIELD. The code implements a method IF_SADL_EXIT_CALC_ELEMENT_READ~CALCULATE. It declares two tables: LT_ADDFIELD and LT_FINAL. LT_ADDFIELD is a standard table of ZC_RAP_COURSE_02 with a default key. LT_FINAL is also a standard table of ZC_RAP_COURSE_02 with a default key. A local variable LT_FINAL is defined as a standard table of ZC_RAP_COURSE_02 with a default key. The code uses a reduce loop to calculate a mark for each row in LT_ADDFIELD based on project and ID. If the mark is not initial, it initializes LT_FINAL with fields from LS_FIELD and sets the REMARK field based on the project being initial or not. The REMARK field is set to 'Excellent' if the project is initial, and 'Good' otherwise. Finally, it returns the calculated data from LT_FINAL. On the right, the 'Variables' view shows the state of the variables. LT_FINAL contains one row with ID 6AE38983FB421EDFAD8E9, Placecompany ABCD, Project XYZ, FRESULT PASS, LASTCHANGEDAT 0.0000000, and REMARK Excellent. ME is set to 0. SY-SUBRC is 0. Locals is set to {O:628*(CLASS=ZCL_ADD...). A checkmark is placed next to the line 'IF lv_mark IS NOT INITIAL.'

```
CLASS zcl_addfield IMPLEMENTATION.
METHOD if_sadl_exit_calc_element_read~calculate.

DATA: lt_addfield TYPE STANDARD TABLE OF zc_rap_course_02 WITH DEFAULT KEY,
      lt_final    TYPE STANDARD TABLE OF zc_rap_course_02 WITH DEFAULT KEY.

lt_addfield = CORRESPONDING #( it_original_data ).

DATA(lv_mark) = REDUCE char1( INIT lv_check TYPE char1
                           FOR <ls_mark> IN lt_addfield
                           NEXT lv_check = COND #( WHEN <ls_mark>-Id IS NOT INITIAL
                                     AND <ls_mark>-project IS NOT INITIAL
                                     THEN abap_true
                                     ELSE abap_false ) ).

✓ IF lv_mark IS NOT INITIAL.

lt_final = VALUE #( FOR ls_field IN lt_addfield
                     ( Id           = ls_field-id
                     Placecompany = ls_field-Placecompany
                     Project      = ls_field-Project
                     Lastchangedat = ls_field-Lastchangedat
                     Fresult       = ls_field-Fresult
                     Remark        = COND #( WHEN ls_field-Project IS NOT INITIAL
                                     THEN 'Excellent'
                                     ELSE 'Good' ) ) ).

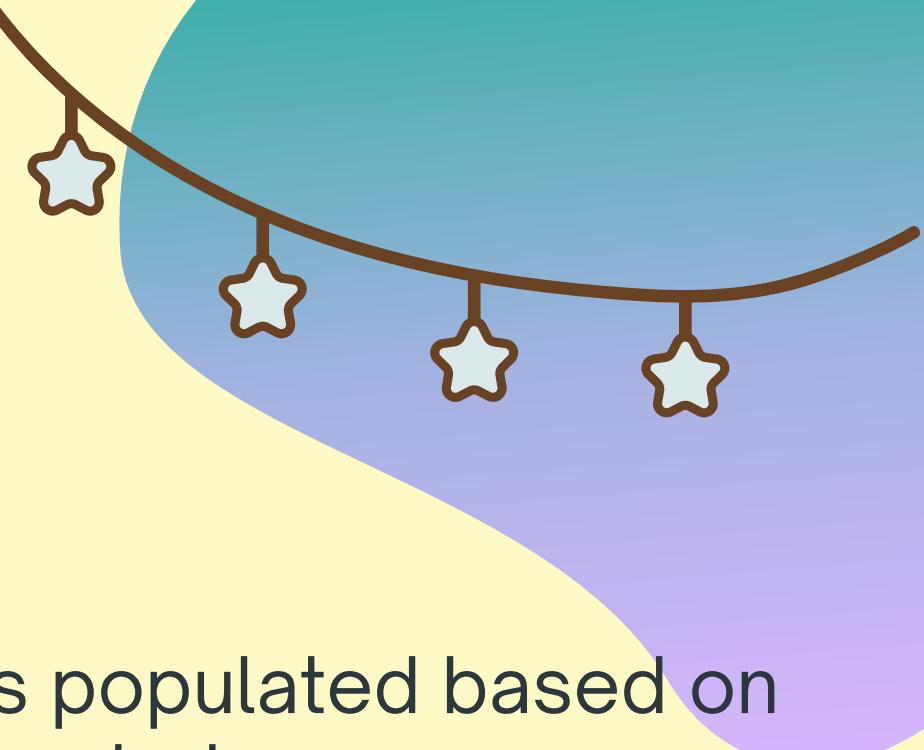
ct_calculated_data = CORRESPONDING #( lt_final ).
ENDIF.
```

Step 6: After providing all other fields. Remark field value will populate using the custom implemented logic in class.

The screenshot shows a SAP Fiori application. At the top, there are tabs for 'Student' and 'Academic Details'. The 'Student' tab is active. Below the tabs, there are input fields for Student ID (6ae38983-fb42-1edf-ad8e-9faead9b9131), First Name (Rupesh), Last Name (Kadge), Status (checkbox), Age (28), and DOB (Dec 14, 2020). Below these fields is a section titled 'Academic Details (1)' with a table. The table has columns: Placed Company, Project, Result, and Result. The first row shows ABCD in the Placed Company column, XYZ in the Project column, and PASS in the Result column. There are buttons for Search, Create, Delete, and other actions at the bottom of the table.

Placed Company	Project	Result	Result
ABCD	XYZ	PASS	>

Rupesh Kadge
SAP Technical Consultant



Step 7: Finally the newly added field is populated based on provided logic and will appear as shown below.

Student ID: 6ae38983-fb42-1edf-ad8e-9faead9b9131	Last Name: Kadge	Status: No
First Name: Rupesh	Age: 28	DOB: Dec 14, 2020
Academic Details (1)		
Placed Company ABCD	Project XYZ	Result PASS





Rupesh Kadge
SAP Technical Consultant



Thank You **& Stay** Tuned.



SAVE AND
SHARE