Rupesh Kadge
SAP Technical Consultant

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.

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

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  }
```

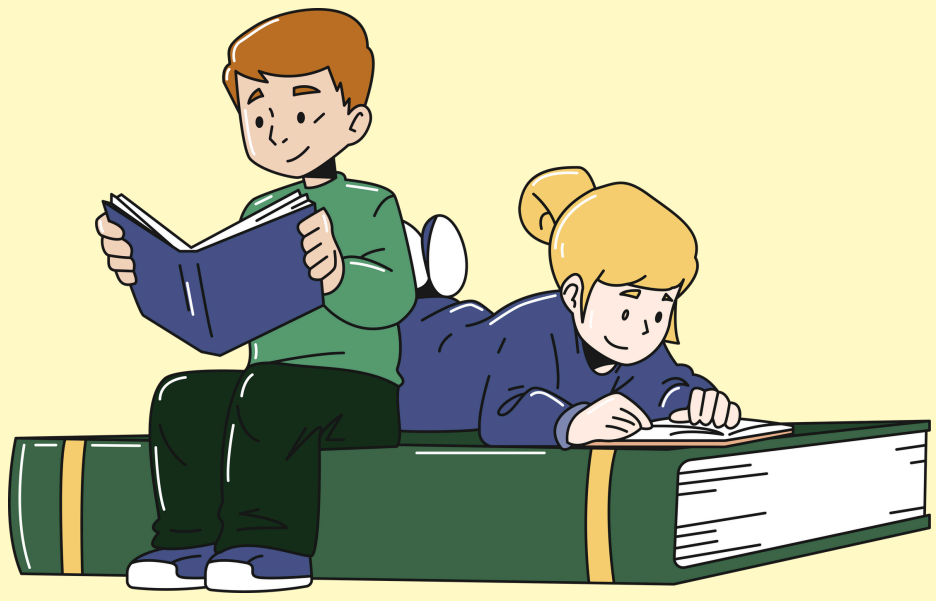
Step 3: The class you added in annotation has to use interface to populate additional field value.

```
CLASS zcl_addfield DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .

  PUBLIC SECTION.
    INTERFACES if_sadl_exit_calc_element_read.

  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.



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

  ENDMETHOD.

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.

```
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.
```



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.

```
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.
```

Variables

Name	Value
<Enter variable>	
LV_MARK	X
LT_FINAL	[1x6(348)]Standard Table
[1]	Structure: flat, not charlike.
ID	6AE38983FB421EDFAD8E
PLACECOMPANY	ABCD
PROJECT	XYZ
FRESULT	PASS
LASTCHANGEDAT	0.0000000
REMARK	Excellent
SY-SUBRC	0
ME	{O:628*\CLASS=ZCL_ADD
Locals	

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

Student | Academic Details

Student ID: 6ae38983-fb42-1edf-ad8e-9faead9b9131
Last Name: Kadge
Status:
First Name: Rupesh
Age: 28
DOB: Dec 14, 2020

Academic Details (1)

Search | Create | Delete

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)

Search

Placed Company	Project	Result	Result
ABCD	XYZ	PASS	Excellent

Rupesh Kadge
SAP Technical Consultant

Thank You & Stay Tuned.

SAVE AND SHARE