

RAP Basics add Delete, Update, Create

Create a *Managed Behavior* for your CDS only for Delete

- Right click on your CDS and select **New Behavior Definition**

New Behavior Definition

Behavior Definition
Create Behavior Definition

Project: *

Package: *

☐ Add to favorite packages

Name:

Description: *

Original Language:

Root Entity: *

Implementation Type: *

- Set the *alias* to **Customer*
- Remove the **create** and **update** options:

```
managed implementation in class zbp_ws##_cds_rap_basic unique;

define behavior for ZWS##_CDS_RAP_BASIC alias Customer
persistent table ZWS##_DT_CUST
lock master
authorization master ( instance )
//etag master <field_name>
{
  delete;
}
```

- Activate the Behavior
- Use **CTRL + SHIFT + 1** to open the *Quick Assist* tab.
- Next click on the Class name in the Behavior and see the proposal in the *Quick Assist* tab.
- Double click the Proposal to generate the class

```

CLASS zbp_ws##_cds_rap_basic DEFINITION PUBLIC ABSTRACT FINAL FOR
BEHAVIOR OF zws##_cds_rap_basic.
ENDCLASS.

```

```

CLASS zbp_ws##_cds_rap_basic IMPLEMENTATION.
ENDCLASS.

```

```

CLASS lhc_Customer DEFINITION INHERITING FROM
cl_abap_behavior_handler.
    PRIVATE SECTION.

        METHODS get_instance_authorizations FOR INSTANCE AUTHORIZATION
            IMPORTING keys REQUEST requested_authorizations FOR Customer
            RESULT result.

```

```

ENDCLASS.

```

```

CLASS lhc_Customer IMPLEMENTATION.

```

```

    METHOD get_instance_authorizations.
    ENDMETHOD.

```

```

ENDCLASS.

```

- Test the delete function from either the preview from Eclipse or the BAS application.
- There is now a **Delete** button, when you select one or more lines you can delete them. Check out the difference between the V2 and V4 applications.
- Or click on a line and use the **Delete** button in the Object Page.

If you have deleted all your records, just run the Class **ZWS##_INIT_DATA** again with F9.

The screenshot shows a web application interface for managing customers. At the top, there is a 'Go' button and a link to 'Adapt Filters'. Below this is a table titled 'Customers (5)'. The table has columns for ID, Name, Street, City, Country, and Email. Five customers are listed: Pixel Tech, Visions, Techaholic, Monotech, and Techlanch. The 'Delete' button in the top right corner of the table is highlighted with a red box. Below the table, a modal dialog titled 'Delete' is displayed, asking 'Delete the selected objects?'. The dialog has a blue 'Delete' button and a grey 'Cancel' button. Below the dialog, a detailed view of a customer (ID: 4, Name: Monotech) is shown. The 'Delete' button in the top right corner of this view is also highlighted with a red box.

ID	Name	Street	City	Country	Email
1	Pixel Tech	Rue de la Meuse 12	Arion	LU	info@Pixel.lu
2	Visions	Rue Thomas Edison 56	Luxembourg	LU	mail@visions.lu
3	Techaholic	De dam 15	Amsterdam	NL	post@techaholic.nl
4	Monotech	Schoolstraat 1	Alphen	NL	po.dep@monotech.nl
5	Techlanch	Bronstraat 6	Brussels	BE	info@techlanch.be

Delete

Delete the selected objects?

Delete Cancel

Customer

ID: 4
Name: Monotech
Street: Schoolstraat 1
City: Alphen
Country: NL
Email: po.dep@monotech.nl

Delete Ctrl+Delete

Add *Edit* Option to the Behavior Definition

- Add the **update** option to the Behavior Definition.

```
managed implementation in class zbp_ws##_cds_rap_basic unique;

define behavior for ZWS##_CDS_RAP_BASIC alias Customer
persistent table ZWS##_DT_CUST
lock master
authorization master ( instance )
//etag master <field_name>
{
    update;
    delete;
}
```

- Activate the Behavior Definition.
- Test the application again. See the difference between the V2 and V4 versions.

The V4 is not displaying the **Edit** Button whereas the V2 is. This is because for the V4 version we need to add the **Draft** option. We will do this later. For now just test with the V2 application.

- When you press the edit button you are able to change the *key* field **ID**. That is not what we want.
- To prevent this we add **field (readonly) ID;** to the Behavior Definition. This will make the ID field read only.

- Activate and the V2 application again.

Add *Create* Option to the Behavior Definition

- Add the **create** option to the Behavior Definition.

```
managed implementation in class zbp_ws##_cds_rap_basic unique;

define behavior for ZWS##_CDS_RAP_BASIC alias Customer
persistent table ZWS##_DT_CUST
lock master
authorization master ( instance )
etag master Lchg_Date_Time
{
    create;
    update;
    delete;

    field (readonly) ID;
}
```

- Activate the Behavior Definition.
- Test the application again.
- You now see that you can only *once* add a new record, because the second new record that you want to make gives you an error:

Other Messages

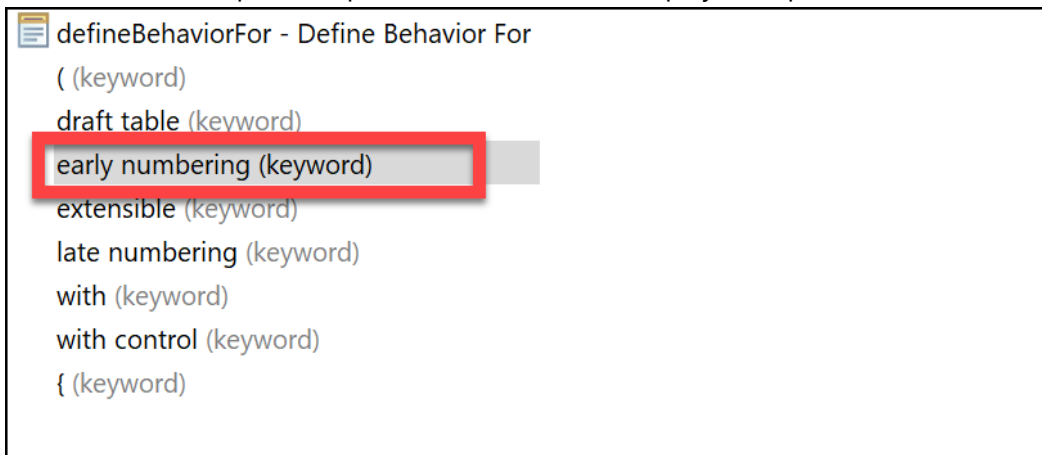


The key value is already in use. Please enter a different one.

- This can be solved by using **early numbering**

Add **early numbering** to the Behavior Definition

- Add the **early numbering** to the Behavior Definition
- Use the Code Completion option **CTRL+SPACE** to display the options



- Double click the word **create** to use the *Quick Assist* to generate a new method
- Implement the new method by checking the MAX number in the table and adding 1 to that max number.

```
METHOD earlynumbering_create.
```

```
    LOOP AT entities INTO DATA(entity) WHERE id IS NOT INITIAL.
        APPEND CORRESPONDING #( entity ) TO mapped-customer.
    ENDLOOP.
```

```
DATA(entities_without_id) = entities.
DELETE entities_without_id WHERE id IS NOT INITIAL.
```

```
"Get max travel ID from standard table
SELECT SINGLE FROM zws##_dt_cust FIELDS MAX( id ) INTO
@DATA(max_cust_id).
```

```
"Set Customer Id
LOOP AT entities_without_id INTO entity.
    max_cust_id += 1.
    entity-Id = max_cust_id.
```

```
APPEND VALUE #( %cid = entity-%cid
                %key = entity-%key
```

```
        ) TO mapped-customer.  
    ENDLOOP.  
  
ENDMETHOD.
```

- Activate the Behavior Definition.
- Test the application again.
- You now see that when you press save the record gets the next number in the table.