A screenshot of a computer screen

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedIn this video tutorial, I will discuss with you about the implementation of validations in Abap RESTful

Application Programming model.

I suggest that you view the video lectures on the topics of business object, transactional, buffer

and entity manipulation language prior to proceeding with this video lecture.

Because in those video lectures, we have covered all the basics that are important to understand the

implementation of validations.

Okay, so let's start.

In the context of a SAP Abap RESTful application programming model.

If validation is an optional part of business object behavior, it is used to check the consistency

of business object instances based on certain trigger conditions.

Trigger conditions can be create, update, delete operations and modified fields.

When these trigger conditions are fulfilled, the validation is implicitly invoked by the Business Objects

framework.

This means that the system automatically checks whether the data in the business object instances is

consistent or not.

One key thing to remember is that validations are always invoked during the same sequence at the end

of business object transaction.

And invoked validation can reject inconsistent instance data from being saved and return message to

the consumer.

In short, a validation plays a crucial role in maintaining the integrity of your data in the RESTful

application programming model.

It ensures that only consistent data is saved, thereby preventing the potential issues down the line.

In today's exercise, we are going to define and implement validation on field flight date.

We will put a validation of flight date so that a new flight booking record instance will not get created

if the flight date is backdated and system will give the error message.

As you can see in this example, assuming the system data has fixed May 2024 during the new flight booking

record creation, the entered flight date here is 3rd May 2024.

So we got an error message saying flight date cannot be parsed dated.

And this new record is not saved in database.

To implement this validation, we need to make changes in CDs behavior definition and its implementation.

As you can see here.

Now let's go to the Abap development tool in eclipse to implement this example.

Okay, so till this point we have developed a Fury application using RESTful application programming

model.

Where we can display the data and perform the Crud operations like create, update and delete.

Now let's see how we can add optional part of business object behavior, such as validations in this

application.

Let's go to Abap development Tools in eclipse.

Here.

We need to get into behavioral definition.

And here we need to add keyword validation.

Then the validation name keyword on save.

And then inside the bracket you need to specify the field name on which he wants to perform the validation.

You can also specify the standard operations create, update and delete.

As I have mentioned earlier, the validations are always invoked during the save sequence and that's

why you need to use this keyword on save.

So here in this bracket we have specified the trigger conditions.

During unsafe sequence this validation will get performed.

And now let's implement this validation.

So as you can see here we got this yellow sign validation is not implemented.

So click here.

Double click.

So this will add a method.

To the behavior implementation class that we have already generated.

So now let's add logic over here.

So here to save your time.

I have already written a logic and I'll be using that.

This logic I have provided here with detailed comments, so that you can easily understand the working

of each statement used in this program.

So here.

Now let's try to understand the written logic.

Let's activate this program.

And set a breakpoint on statement read.

Entities.

Now, let's go to your application.

Since we wanted to do the validation on the flight date during the new record creation.

Hit the create button.

Let's provide here.

Airline ID as a connection.

ID as 9999.

Booking ideas four times eight.

Here.

Let's get the flight date.

As you can see here, the system date is 19th May 2024.

I will give a back date 16th May.

I will do passenger name as P name one.

Okay.

So all this information that we have provided on your screen forms the interaction phase of Business

Object Runtime.

And when we hit the button create, all these information will get saved to transactional buffer and

then will get saved to database.

And we know the validation on flight date will triggered during the save sequence.

So let's hit the create button here.

Since we have set a breakpoint.

The program control has reached here.

So as you can see here, control has reached to the method validate flight date.

Since we want to do the validation on flight date, we need to read the content that has been supplied

on US screen in our program.

For that purpose, we will use the entity manipulation language statement read entities off.

We use entity manipulation language to read and modify content in transactional buffer.

After keyword read entities of you need to provide the root entity name.

In local mode is used to exclude authorization checks.

After keyword entity, you need to specify the entity name from which you want to read the content.

In our example, our business object contains only root entity.

That's why.

If both name are same.

Then here we want to read all the fields from this entity for the corresponding fields which are present

in this derived internal table.

Keys.

Which contains information related to key fields.

As you can see here that has been passed on the screen.

And the result will get collected in this internal table.

So let's execute this statement.

Now let's see the content in this internal table.

You can see here we got values against each key field.

And then the other field information, such as passenger name that we have passed on screen.

So basically using these three statement, we have simply read the record that we have passed on us

screen.

Here we are looking at the content present in this internal table to check if the flight date pass on

the UI screen is less than the system date.

If so, then we need to stop that record being saved on database, and we need to raise message.

And we need to raise the error message saying the flight date cannot be passed a date.

So here the flight date is 16th, which is less than the system date.

System date is 19th May.

So this if condition becomes true.

So in this case, validation is failed and we want to stop that record being saved on database.

To do so, we need to pass.

The key field of field instance.

2D film structure.

Okay.

Let's execute this statement.

Here, authentic is a derived type component group.

Summarizing all the prime key fields of above object instance, as you can see here.

It contains all the key fields and their values that we have passed on UI screen.

We need to pass this information to the failed structure here.

Failed is used for logging the instance for which operation has failed.

Freddy is a deep structure which has one component, which is of name of the entity, which is a standard

table.

The inside it you can see.

The key fields and the values are populated.

You can see the key fields along with some other derived fields.

So using this statement, we have passed the key fields of record to these fields of structure.

Field.

After execution of this statement system will not update the record in database table.

Now the next step is we need to populate the message on front end saying that flight date cannot be

overstated.

To do so, we need to pass information to structure reported.

Now let's execute this statement.

Now let's look at this structure reported.

This is a deep structure which contains component with the name of our entity.

Which is a standard table.

And then we have a key fuse along with some other fields.

To display message on screen.

We need to pass some information to this field.

Percent MSG.

Here.

We need to pass the text that we want to display on screen.

And then message type.

And we are doing it using this method.

Call new message with text and by passing the values to these parameters.

Severity and text.

Percent element is used to assign masses to the target fields by marking the data field in the percent

element structure.

This interface contains information zero one.

That we are passing to the flight date of percent element.

So if you look at the deep structure reported, you have the component name element.

The structure element contains all the fields that are there in entity, out of which we need to highlight

flight date.

And that's why we are passing value zero one.

Okay.

That's it.

This.

This failed structure is used to stop the record being saved in database.

This structure is used to populate message on UI screen.

Now let's do a fit.

And go back to our Fury application.

So as you can see, this flight date field is highlighted and the message has been popped up saying

flight date cannot be passed.

Right flat.

It can now be stated.

Now instead of 16.

Let's make it 20.

Then hit the create button.

At this time.

There's two effect.

Since flight date is greater than the system date, we didn't get any error message.

And let's check if this record is saved in database.

Passenger name.

Use B name one.

Okay.

So as you can see, the record has been saved on database.

Okay, so with this example now I hope you understood the working of validation in principle application

programming model.

Thank you for watching this video.

Have a nice day ahead.