Offline Help Files

6.04.02 SynR3.js Global Objects

There are four global objects that are defined in the SynR3.js file. These are as follows:

- SR3Connection
- SR3Transaction
- SR3TransactionObject
- SR3Screen SR3Connection

SR3Connection

The SR3Connection object is used to represent a client connection. Taking the name of the connection as a string argument in the constructor, this object can be used to manage connections to SAP. The syntax to use the SR3Connection object is as follows:

```
SR3Connection(strConnectionName );
```

In the following example, we are creating a new connection:

```
objConnection = new SR3Connection("test server");
```

The SR3Connection object has four attached methods, which will be defined and explained in the following sections.

onCreateConnection

The onCreateConnection method is called by the framework when a new connection is created. The syntax is as follows:

```
SR3Connection.onCreateConnection( );
```

See the following example:

```
objConnection.onCreateConnection( );
```

onDeleteConnection

The onDeleteConnection method is called by the framework when an existing connection is deleted. The syntax is as follows:

```
SR3Connection.onDeleteConnection( );
```

See the following example:

```
objConnection.onDeleteConnection( );
```

onStartTransaction

The onStartTransaction method is called by the framework when a new connection is created. The requisite parameters are the transaction code, the screen number, the message string and setting the screen refresh to true or false. These parameters are passed as arguments and the syntax is as follows:

```
SR3Connection.onStartTransaction(strTransactionCode,nScreenNumber,bRef
reshScreen,strMessageString );
```

See the following example:

```
objConnection.onStartTransaction(strTransactionCode,nScreenNumber,bRef
reshScreen,strMessageString );
```

onProcessTransaction

The onProcessTransaction method is called by the framework when a new connection is created. The syntax is as follows:

```
SR3Connection.onProcessTransaction( );
```

See the following example:

```
objConnection.onProcessTransaction( );
```

SR3Transaction

The SR3Transaction object is used to create and manage transactions within an Offline session or connection. The constructor takes the following parameters as arguments:

- Connection name
- Transaction code
- Screen number
- Refresh screen Boolean (true or false)
- Message string

The syntax is as follows:

```
SR3Transaction(strConnectionName,strTransactionCode,nScreenNumber,bRef
reshScreen,strMessageString );
```

The following example shows how to create a new Transaction object in Offline:

```
objTransaction = new SR3Transaction(strConnectionName,strTransactionCo
de,nScreenNumber,bRefreshScreen,strMessageString );
```

The SR3Transaction object has seven methods attached, which will be explained in the following sections.

onInitializeTransaction

The onInitializeTransaction method is called when a user starts a new transaction. The requisite parameters are the transaction code, the screen number, the message string and setting the screen refresh to true or false. These parameters are passed as arguments and the syntax is as follows:

```
SR3Transaction.onInitializeTransaction(bRefreshScreen,strMessageString
 );
```

See the following example:

```
objTransactionl.onInitializeransaction(bRefreshScreen,strMessageString
 );
```

onEnterTransaction

The onEnterTransaction method is called immediately following the user's initialization of a new transaction. The requisite parameters are the array containing the status strings, the message string and setting the screen refresh to true or false. These parameters are passed as arguments and the syntax is as follows:

```
SR3Transaction.onEnterTransaction(arStatusScreens,bRefreshScreen,strMe
ssageString);
 See the following example:
objTransaction.onEnterTransaction(arStatusScreens,bRefreshScreen,strMe
ssageString);
```

sendScreen

The sendScreen method is used to send the current transaction screen based on the screen number. The requisite parameters are the screen number, the message string and setting the screen refresh to true or false. These parameters are passed as arguments and the syntax is as follows:

```
SR3Transaction.onStartTransaction(nScreenNumber,bRefreshScreen,strMess
ageString);
 See the following example:
objTransaction.onStartTransaction(nScreenNumber,bRefreshScreen,strMess
ageString);
```

sendPreviousScreen

The sendPreviousScreen method is used to send the previous screen of the user's current transaction. The only parameter required is the screen refresh Boolean value. The parameter is passed as an argument and the syntax is as follows:

```
SR3transaction.sendPreviousScreen(bRefreshScreen);
```

See the following example:

```
 objTransaction.sendPreviousScreen(bRefreshScreen);
```

sendNextScreen

The sendNextScreen method is used to send the screen immediately following the current screen in the user's current transaction. The only parameter required is the screen refresh Boolean value. The parameter is passed as an argument and the syntax is as follows:

```
SR3transaction.sendPreviousScreen(bRefreshScreen);
```

See the following example:

```
objTransaction.sendPreviousScreen(bRefreshScreen);
```

onLeaveTransaction

The onLeaveTransaction method is called by the framework when the user leaves the current transaction. There are no parameters required for this method. The syntax is as follows:

```
SR3Transaction.onLeaveTransaction(  );
```

See the following example:

```
objTransaction.onLeaveTransaction(  );
```

cleanScreen

The cleanScreen method is called by the framework when a new connection is created. The requisite parameter is the Boolean value for the screen refresh. This parameter is passed as an argument and the syntax is as follows:

```
SR3Transaction.cleanScreen(bRefreshScreen);
```

See the following example:

```
objTransaction.cleanScreen(bRefreshScreen);
```

SR3TransactionObject

The SR3TransactionObject is used to represent an individual transaction in Offline. The constructor takes the Transaction code string as a parameter when the object is created. The syntax is as follows:

```
SR3TransactionObject(strTransactionCode);
```

The example below we are creating a new transaction object based on transaction 'MM01':

```
var strMM01TCode = "MM01";
objTransaction = new SR3TransactionObject(strMM01TCode);
```

We set the transaction code as a variable, enabling us to pass it to the method more easily and also making it portable for later use if needed.

SR3TransactionObject has four attached methods, which we will describe and demonstrate in the following section.

addDatabaseTable

The addDatabaseTable method is used to add a database definition to the current transaction. The database definition is passed as an argument to the constructor and the syntax is as follows:

```
SR3Transaction.addDatabaseTable(arDatabaseDefinition);
```

See the following example:

```
objTransaction.addDatabaseTable(arDatabaseDefinition)
objXX01.addDatabaseTable( [{
fieldname:"g_first_name",columntitle:"First Name",cvolumnwidth:16,fiel
dlength:16,fieldtype:DBF_STRING,keytype:DBT_NON_PRIMARY_KEY,columntype
:CTRL_TABLE_STATIC} ] );
```

addScreen

The addScreen method is used primarily when a user wishes to add a new screen definition to the current transaction. The array containing the screen definitions is passed as an argument and the syntax is as follows:

SR3Transaction.addScreen(arScreenDefinition);

See the following example, where we are adding a new screen:

```
objTransaction.addScreen(arScreenDefinition);
objXX01.addScreen([
{type:SCR_CAPTION,label:"TRIAL"},
{type:SCR_EDIT,label:"First
Name",position:{
row:2,col:1,edcol:14,edlen:16},
fieldname:"g_first_name",maxtextlength:16,required:1}]);
```

addNavigationScreen

The addNavigationScreens method is used primarily when a user wishes to add a new navigation screen definition to the current transaction. The array containing the screen definitions is passed as an argument and the syntax is as follows:

```
SR3Transaction.addNavigationScreens(  );
```

See the following example, where we are adding a new screen:

```
objTransaction.addNavigationScreens(  );
objMenu.addNavigationScreens( [
{type:SCR_CAPTION,label:"Offline Main Menu"},
{type:SCR_GROUPBO
X,label:"Offline Main Menu",posit
ion:{row:0,col:0,height:9,width:25},fieldname:"z_menu_group},
{type:SCR_PUSHBUTTON,la
bel:"@0X@Logoff",postion:{row:7, col:1,width:20},
fieldname:"FWKDR",fcode:"/next"} ] );
```

createTable

The createTable method is used to create a database table for a given transaction. The field name and the field elements object are passed as arguments to the constructor and the syntax is as follows:

```
SR3Transaction.createTable(strTableName,objFieldElements);
```

See the following example:

```
objTransaction.createTable(strTableName,objFieldElements("table name",
objFieldElements);
```

SR3Screen

The SR3Screen object is used to represent an individual screen in a given transaction. The parameters needed are the screen definition array, the main modal screen object and the status message string. These are passed to the constructor as arguments. The syntax is as follows:

```
SR3Screen(objCurrentTransaction,arScreenDefinitions,objMainModalScreen
,bRefreshScreen,strMessageString );
```

See the following example:

```
objScreen = new SR3TransactionObject(objCurrentTransaction,arScreenDef
initions,objMainModalScreen,bRefreshScreen,strMessageString );
```

The SR3Screen object has eleven methods attached. These methods are described and demonstrated in the following sections.

getMultiLangLabelText

The getMultiLangLabelText method is get the multi-lingual label text for a given transaction. This method can be written in two different manners, depending if the method arguments are the label definition string or the array containing the label definitions. If the label definition string is used, the syntax is as follows:

```
SR3Screen.getMultiLanguageLabelText(strLabelDefinition);
```

If the array for the label definitions is used, the syntax would be as shown below:

```
SR3Screen.getMultiLanguageLabelText(arLabelDefinition );
```

Please see the following example:

```
strLabel = objScreen.getMultiLanguageLabelText("label" );
```

createScreen

The createScreen method uses the screen definition to create a real screen for a given transaction. The syntax is as follows:

```
SR3Screen.createScreen( );
```

See the following example:

```
bResult = objScreen.createScreen( );
```

findTableDefinition

The findTableDefinition method is used to find the definition of a table in the current screen. The syntax is as follows:

```
SR3Screen.findTableDefinition( );
```

See the following example:

```
objTable = objScreen.findTableDefinition( );
```

findListDefinition

The findListDefinition method is similar to the findTableDefinition method previously described. It is used to find the list definition for the current screenThe syntax is as follows:

```
SR3Screen.findListDefinition( );
```

See the following example:

```
objList = objScreen.findListDefinition( );
```

createTableListControl

The createTableListControl method is used to cerate a database table for a given transaction. The field name and the field elements object are passed as arguments to the constructor and the syntax is as follows:

```
SR3Screen.createTable(objTableListDefinition );
```

See the following example:

```
bResult = objScreen.createTable(objTableListDefinition );
```

onEnterScreen

The onEnterScreen method is called immediately when the user enters a enw screen. The requisite parameters are the array containing the status strings, the message string and setting the screen refresh to true or false. These parameters are passed as arguments and the syntax is as follows:

```
SR3Screen.onEnterScreen(bRefreshScreen,strMessage );
```

See the following example:

```
objScreen.onEnterScreen(bRefreshScreen,strMessage );
```

onSendScreen

The onSendScreen method is called whena screen is sent. The syntax is as follows:

```
SR3Transaction.onEnterTransaction(bRefreshScreen,strMessage );
```

See the following example:

```
objScreen.onSendScreen(bRefreshScreen,strMessage );
```

cleanContents

The cleanScreen method is called by the framework when a new connection is created. The requisite parameter is the Boolean value for the screen refresh. This parameter is passed as an argument and the syntax is as follows:

```
SR3Screen.cleanScreen(bRefreshScreen);
```

See the following example:

```
objScreen.cleanScreen(bRefreshScreen);
```

addScreen

The addScreen method is used to add a new scren on top of the current screen. The array containing the screen definitions is passed as an argument and the syntax is as follows:

```
SR3Screen.addScreen(arScreenDefinitions);
```

See the following example, where we are adding a new screen:

```
objScreen.addScreen(arScreenDefinitions);
```

removeTopScreen

The RemoveControl method is used to removethe top screen added by the addScreen method from the current screen. The syntax is as follows:

```
SR3Screen.removeTopScreen(  );
```

In the following example, we are removing a table control from the current screen:

```
obj Screen.removeTopScreen(  );
```

onProcessScreen

The onProcessScreen method is called when a new screen is received. The syntax is as follows:

```
SR3Screen.onProcessScreen(objCurrentTransaction);
```

See the following example:

```
obj Screen.onProcessScreen(objCurrentTransaction);
```

Unique solution ID: #2090
Author: sarvani.kusuri@guixt.com
Last update: 2021-06-03 18:31