```
objSession = new JSession(objConnection);
```

There are four online objects defined in the SynR3.js file that are used for communicating with SAP ERP. These objects are as follows:

- JConnection
- JSession
- JModal
- JServerHost

JConnection

The JConnection object is used to create an online connection with the SAP application server by using the logon informatioon entered by the user. The syntax is shown below:

```
JConnection(strServerName,strClientNumber,strUserName,strPassword,strLanguage);
```

In the following example, we are creating a new JConnection object and specifying the parameters:

```
objConnection = new JConnection("serverName","800","username","password","en");
```

The JConection object has only a single method associated with it, as described below. Close The Close method closes an open connection to the SAP ERP system. It takes no arguments and the syntax is as follows:

```
JConnection.Close( );
```

In the following example, we will close a conection and set the connection object equal to null in order to collect any garbage that may exist:

```
objConnection.Close( );
objConnection = null;
```


(c) 2026 Liquid UI | Synactive | GuiXT <dev@guixt.com> | 2026-11-15 00:47
URL: https://www.liquid-ui.com/knowledge_base/content/137/1090/en/60403-offline-communication-objects.html

JSession

The JSession object is used to create a new session within a given connection to a SAP application server. Taking the connection object as an argument, the syntax is as follows:

```
JSession(objConnection);
```

To create a new session, see the following example:

```
objSession = new JSession(objConnection);
```

There are no methods associated with the JSession object.

JModal

JModal is used to create a modal screen in an online session object. Taking the session object as an argument, the syntax is as follows:

```
JModal(objSession);
```

The following example demonstrates how to create a new JModal object:

```
objModal = new JModal(objSession);
```

The JModal object has five associated methods, which are described below.

GetTransaction

This Boolean object sends a transaction code to SAP and waits to receive a screen in return. Taking the transaction code as an argument, the syntax is as shown below:

```
JModal.GetTransaction(strTrtansactionCode);
```

In the following example, we are checking to see what the transaction object to be

(c) 2026 Liquid UI | Synactive | GuiXT <dev@guixt.com> | 2026-11-15 00:47
URL: https://www.liquid-ui.com/knowledge_base/content/137/1090/en/60403-offline-communication-objects.html

received will be and then performing additional actions based on the result of that test.

```
if(objModal.GetTransaction("zguixt") == true) {
  // additional task logic goes here:
} else {
 // some error handling logic goes here:
 }
```

SetElementValue

The SetElementValue is a Boolean method that sets the value of a given screen element. It takes two arguments - the field name where the value will be stored and also the value to be set. The syntax is as follows:

```
JModal.SetElementValue(strFieldName,strValue);
```

In the following example, we are using this to set a value and then perform additional actions based on a test if the value is true.

```
if(objModal.SetElementValue("z_process","wkdr") == true) {
 // additional task logic goes here:
} else {
 // error handling logic goes here:
 }
```

GetElementValue

This method is a string method that will get the value of a given screen element. It takes the field name as an argument and uses the following syntax:

```
JModal.GetElementValue(strFieldName);
```

The following example will get the value contained in the field 'z_process'. strValue = objModal.GetElementValue("z_process");

Enter

(c) 2026 Liquid UI | Synactive | GuiXT <dev@guixt.com> | 2026-11-15 00:47
URL: https://www.liquid-ui.com/knowledge_base/content/137/1090/en/60403-offline-communication-objects.html

The Enter method is a Boolean that serves as a user-action of pressing the Enter key. It takes no arguments and the syntax is as follows:

```
JModal.Enter( );
```

In the following example, we will test if the user has pressed Enter and then execute some actions based on the results of that test.

```
 if(objModal.Enter( ) == true) {
  // additional task logic goes here:
} else {
 // error handling logic goes here:
}
```

GetMessageString

The GetMessageString method is used to get a message string from a given screen. The syntax is as follows:

```
JModal.GetMEssageString( );
```

In the following example, we will get a particular string and set it equal to a string variable named 'strMessage':

```
strMessage = objModal.GetMessageString( );
```

JServerHost

The JServerHost object is used to designat an Offline server host object. It takes no arguments and teh syntax is as follows:

```
JServerHost( );
```

The following example demonstrates how to create a new JServerHost object:

```
objServerHost = new JServerHost( );
```


(c) 2026 Liquid UI | Synactive | GuiXT <dev@guixt.com> | 2026-11-15 00:47
URL: https://www.liquid-ui.com/knowledge_base/content/137/1090/en/60403-offline-communication-objects.html

JServerHost takes three methods, which are defined in the following sections.

- **IsSignalServerEnd**

  This Boolean methods checks to see if the Offline server is signaling that it will exit. It has no arguments and the syntax is as follows:

  ```
  JServerHost.IsSignalServerEnd( );
  ```

  This is useful to perfom actions based on whether or not the server is signaling that it will end, as shown in the example below:

  ```
  if(objServerHost.IsSignalServerEnd( ) == true){
  // perform additional task logic here:
   }else
   {
   // some error handling logic goes here:
   }
  ```

- **IsSignalToEnd**

  Similar to the IsSignalServerEnd method, this Boolean method checks if the client connection is signaling that it will end. The syntax is as follows:

  ```
  JServerHost.IsSignalToEnd( );
  ```

  This is useful to perfom actions based on whether or not the server is signaling that it will end, as shown in the example below:

  ```
  if(objServerHost.IsSignalToEnd( ) == true) {

  // perform additional task logic here:
  }
  else {
  // some error handling logic goes here:
  }
  ```

- **RunJavaScript**

  The Boolean RunJavaScript method is used to execurte JavaScript functions. It takes a waitflag that specifies if the Javascript is to be run immediately or not and the syntax is as follows:

(c) 2026 Liquid UI | Synactive | GuiXT <dev@guixt.com> | 2026-11-15 00:47
URL: https://www.liquid-ui.com/knowledge_base/content/137/1090/en/60403-offline-communication-objects.html

```
JServerHost.RunJavaScript(iWaitFlag);
```

The WaitFlag parameter has two possible states, as follows:

- 1 = EVENT_WAIT
- 0 = EVENT_NOWAIT

In the example below, we will use the Run JavaScript method to immediately execute a script:

```
objServerHost.RunJavaScript(0);
```

Unique solution ID: #2093
Author: sarvani.kusuri@guixt.com
Last update: 2021-06-03 18:34

(c) 2026 Liquid UI | Synactive | GuiXT <dev@guixt.com> | 2026-11-15 00:47
URL: https://www.liquid-ui.com/knowledge_base/content/137/1090/en/60403-offline-communication-objects.html