

# Measure the Performance of SAP IBP Actions

One advantage of using custom VBA code is that several steps can be combined. Within one button, different simulation runs can be triggered, and data changes can be made and saved. This can be useful in different ways.

## HOW TO START

In our example, we explain how you could set up an upgrade test to measure the performance of different actions in the SAP IBP, add-in for Microsoft Excel (Excel add-in). You can also use the test to make sure that those steps are still working after an upgrade of your SAP IBP system.

In the respective .xslm file for this use case, you find one worksheet with the planning view for which we are running the tests, and a second worksheet including the upgrade test itself, where the performance measurements are collected. See the screenshot below.

Run Upgrade Test		Reset Status	
Test Step	Time Elapsed (sec)		
Refresh			
Simulate			
Run Forecast in Simulation			
Save			

The upgrade test we implemented in this example includes the following steps:

1. Refresh the planning view  
`IBPAutomationObject.Refresh`
2. Change a cell  
`ActiveSheet.Range("K7").Value = 100`
3. Run "Simulate Basic"  
`Call IBPAutomationObject.Simulate`
4. Run a statistical forecast operator in simulation mode  
`Call IBPAutomationObject.Simulate ("My forecast model", "IBPFORECAST")`
5. Save data  
`Call IBPAutomationObject.SaveData (True)`

You can suppress the dialog for reason codes, comments, and sharing, to not interrupt the performance measurements, by setting the parameter for suppressReasonCodeCommentDialog of the API SaveData to "True". For more information about SaveData, see [SaveData](#) on the SAP Help Portal.

To measure the performance of the different steps, the start and end time is collected, and the difference is added to the specific cells in the upgrade test worksheet as shown below.

```
StartTime = Timer
IBPAutomationObject.Refresh
SecondsElapsed = Timer - StartTime
tester.Range("E8").Interior.ColorIndex = 35
tester.Range("E8").Value = SecondsElapsed
```

The whole code looks the following:

```
Private Sub RunUpgradeTest_Click()

    Dim StartTime As Double
    Dim SecondsElapsed As Double
    Dim tester As Worksheet

    ResetStatus_Click

    On Error GoTo ErrorHandler:
    If IBPAutomationObject Is Nothing Then Set IBPAutomationObject =
        Application.COMAddIns("IBPXLClient.Connect").Object
    Set tester = ActiveSheet

    ActiveWorkbook.Sheets("Test 1").Activate
    'refresh
    StartTime = Timer
    Call IBPAutomationObject.Refresh
    SecondsElapsed = Timer - StartTime
    tester.Range("E8").Interior.ColorIndex = 35
    tester.Range("E8").Value = SecondsElapsed

    'change cell and simulate
    ActiveSheet.Range("K7").Value = 100
    StartTime = Timer
    Call IBPAutomationObject.Simulate
    SecondsElapsed = Timer - StartTime
    tester.Range("E9").Interior.ColorIndex = 35
    tester.Range("E9").Value = SecondsElapsed

    'run simulation for forecast model xyz
    StartTime = Timer
    IBPAutomationObject.Simulate "001MoveAvg2", "IBPFORECAST"
    SecondsElapsed = Timer - StartTime
    tester.Range("E10").Interior.ColorIndex = 35
    tester.Range("E10").Value = SecondsElapsed

    'save data
    StartTime = Timer
    Call IBPAutomationObject.SaveData(True)
    SecondsElapsed = Timer - StartTime
    tester.Range("E11").Interior.ColorIndex = 35
    tester.Range("E11").Value = SecondsElapsed
```

```
'show the upgrade test worksheet  
tester.Activate
```

*Exit Sub*

*ErrorHandling:*

*'Implement error handling to help the user to understand what went wrong  
MsgBox Err.Description, vbOKOnly, "Microsoft Excel: Custom VBA code"*

*End Sub*

The ResetStatus\_Click is just cleaning the last results:

```
Private Sub ResetStatus_Click()  
    Range("B8:E16").Interior.ColorIndex = 0  
    Range("E8:G16").Value = ""  
End Sub
```

Feel free to adjust and include the steps that you want to test for your use case. You can run further operators in simulation mode or include further worksheets with planning views which need to be tested. Please keep in mind that the VBA code will have the same effect as clicking the respective buttons in the SAP IBP ribbon. If an error message comes up, the user interaction will be measured as well.

Please note: Code lines which already have been explained in the tutorial for other use cases are not explained in detail again, so please check those as well.

Follow us



[www.sap.com/contactsap](http://www.sap.com/contactsap)

© 2022 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company.

The information contained herein may be changed without prior notice. Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

In particular, SAP SE or its affiliated companies have no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation, and SAP SE's or its affiliated companies' strategy and possible future developments, products, and/or platforms, directions, and functionality are all subject to change and may be changed by SAP SE or its affiliated companies at any time for any reason without notice. The information in this document is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, and they should not be relied upon in making purchasing decisions.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

See [www.sap.com/trademark](http://www.sap.com/trademark) for additional trademark information and notices.