Password Change/Reset Tool in Webdynpro Java

Applies to:

SAP Netweaver 6.0, SAP Netweaver 7.0, SAP Enterprise Portal. For more information, visit the Webdynpro Java Homepage

Summary

The article provides the logic and flow to build a customize password reset and change application in Webdynpro Java. It also provides the sample code to facilitate the development. The application is tested in SAP NW 7.0 SP9 but it can be implemented on earlier or later versions of SAP NW.

Author: Abhinav Sharma

Company: HCL Technologies Ltd.

Created on: 5th January 2011

Author Bio

Abhinav Sharma is SAP certified NetWeaver Consultant. His expertise is in Webdynpro Java/ABAP and SAP EP.

Table of Contents

Introduction

This article provides an idea to implement a custom based self service password reset and change tool where SAP standard solution cannot be used. SAP standard self service tool works if we use SAP EP UME, however, if SAP EP is configured with the SAP R/3 system (Table USR01), then a customize self-service application is required to enable users to manage their own passwords.

Pre-requisite

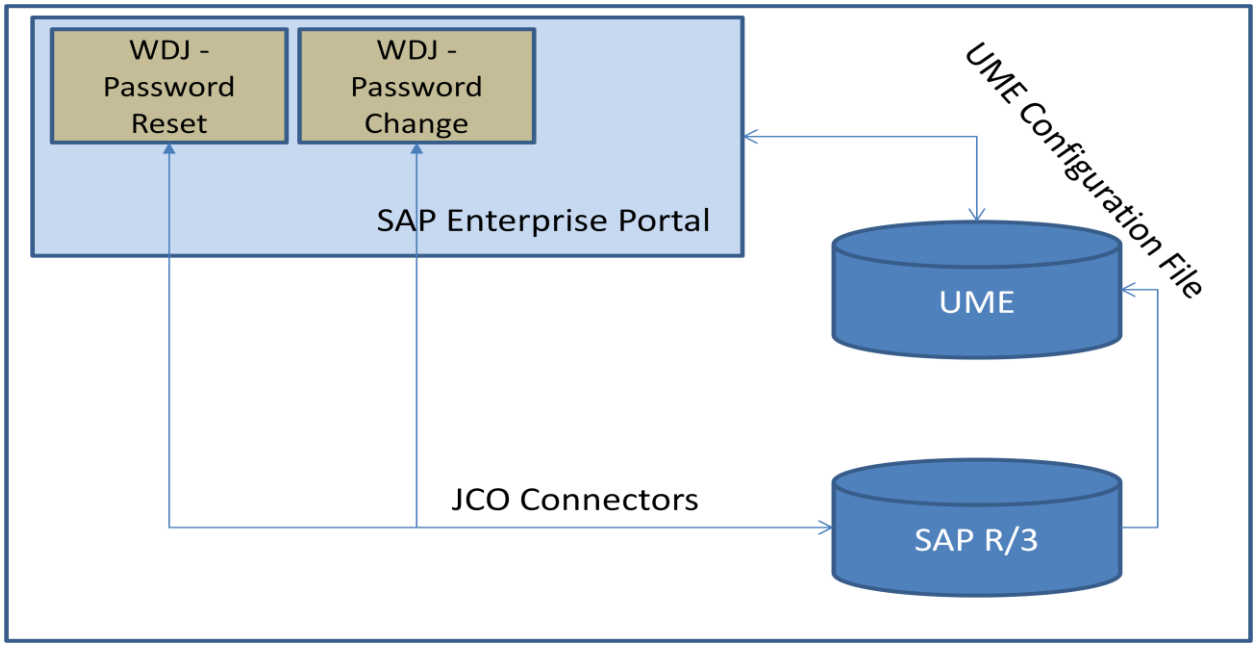
1. Experience in developing applications in WebDynpro Java
2. Understanding of RFCs and its usage in WebDynpro.
3. Knowledge of SAP Portal desirable.

Overview of Application

Most of the time, users or developers sought BASIS help to reset or change the password. The activity can be time consuming depending upon the availability of BASIS team. By automating this process and by allowing users to manage their own passwords, lot of efforts can be saved. There are so many cases when user or developer has to wait to reset their password due to several reasons. This simple tool will help the developers or users to manage their passwords in more effective manner. This will not only reduce the effort but also provide the flexibility to users to change their passwords whenever required. Moreover, the application is compliant with the password policy of the backend system. It is based upon the simple principle and adheres to the password policies of respective backend system.

Application Architecture

The application has very straightforward architecture. Password Change and Password Reset applications developed in Webdynpro Java and interacts with the SAP R/3 System using RFCs. Password Change application can be accessible once the user logged in to the portal, however, a separate URL has been provided to the user to reset their passwords on the SAP Portal Login Screen.



Once user has decided to change the password or reset the previous password, RFC gets executed in the backend system and updates the USR01 table.

Implementation

Overview

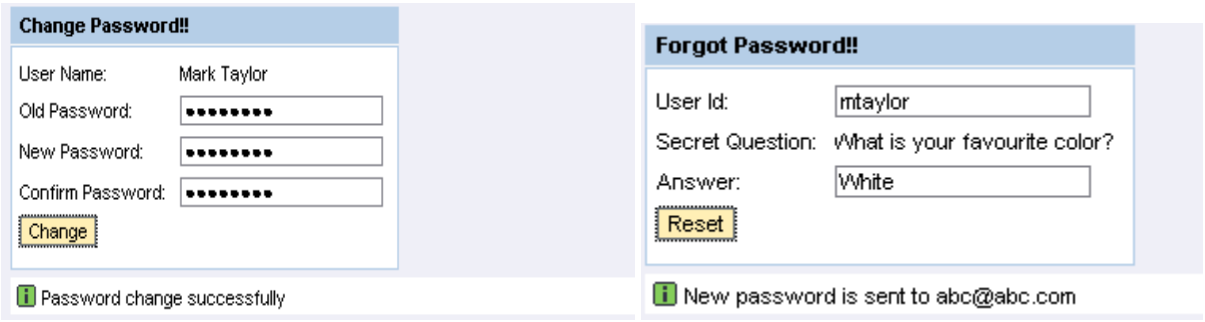
The tool is very simple to use. If user has forget the password, he/she clicks on the forget password link on the login page. To add the forget password URL on the SAP EP login page, refer Logon Screen Customization. Once user has clicked the forget password link, he is required to answer the secret question and upon successful verification, an email has been sent to the registered email account.

Similarly, user can change the password by clicking on Change Password application.

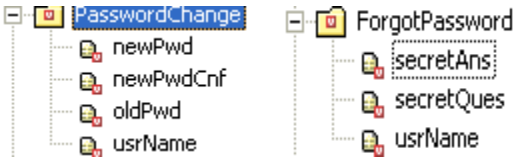
The RFCs that is used to reset and change the passwords are SUSR_USER_CHANGE_PASSWORD_RFC and BAPI_USER_CHANGE. After filling up the required parameters, these RFCs update the record in the USR02 table of the backend system.

Screen Layout and Application Design

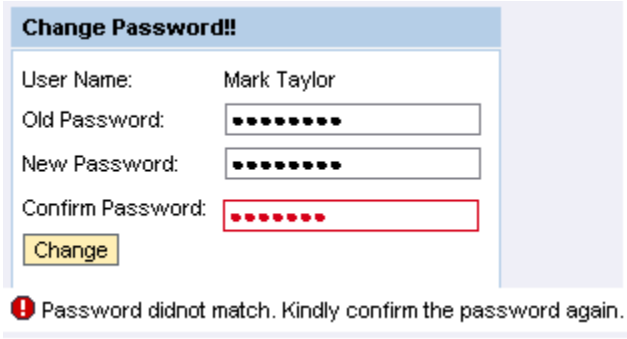
The layout of the screen is designed as shown below.



These forms are bound to below mentioned value attributes



Once the user clicks on button Change and Reset, two methods gets called, mChangePassword() and mResetPassword(). Before calling mChangePassword(), the system verifies if New Password and Confirm Password matches. If not, error message is displayed on the screen.



To enforce the password policy PASSWORD_FORMAL_CHECK RFC can also be called in the validation method.

Sample Code

Parameters UsrName, OldPwd and NewPwd are required to execute the RFC. Use_Bapi_Return, enables the RFC to return the messages in the BAPIRET2 structure

mChangePassword() -

```
wdContext.nodeSusr_User_Change_Password_Rfc_Input().invalidate();
          try {
                short bapiReturn = 1;
                short newException = 0;

                Susr_User_Change_Password_Rfc_Input pwdChnge =
                      new Susr_User_Change_Password_Rfc_Input();
                pwdChnge.setBname(
                      wdContext.currentPasswordChangeElement().getUsrName());
                pwdChnge.setPassword(
                      wdContext.currentPasswordChangeElement().getOldPwd());
                pwdChnge.setNew_Password(
                      wdContext.currentPasswordChangeElement().getNewPwd());
                pwdChnge.setUse_Bapi_Return(bapiReturn);
                pwdChnge.setUse_New_Exception(newException);
wdContext.nodeSusr_User_Change_Password_Rfc_Input().bind(
                                                      pwdChnge);

                wdContext
                      .nodeSusr_User_Change_Password_Rfc_Input()
                      .currentSusr_User_Change_Password_Rfc_InputElement()
                      .modelObject()
                      .execute();

                IPrivateChangepasswordView.IReturn_Change_PasswordElement retMsg =
                      wdContext
                            .nodeReturn_Change_Password()
                            .currentReturn_Change_PasswordElement();
                if (retMsg != null && retMsg.getType().equalsIgnoreCase("E")) {
                      wdComponentAPI.getMessageManager().reportException(
                            "Error Message - "
                                  + retMsg.getMessage(),
                            false);
                } else if (
                      retMsg != null && retMsg.getType().equalsIgnoreCase("S")) {
                      wdComponentAPI.getMessageManager().reportSuccess(
                            "Success - " + retMsg.getMessage());
                } else {
                      wdComponentAPI.getMessageManager().reportWarning(
                            "Warning - " + retMsg.getMessage());
                }

          } catch (WDDynamicRFCExecuteException e) {
                wdComponentAPI.getMessageManager().reportException(
                      "Exception - " + e.getMessage(),
                      false);

                e.printStackTrace();
```

Similarly, in method mResetPassword(), RFC BAPI_USER_CHANGE input is called. The only parameter thatneeds to be filled is UserName and Bapipwd structure.

mResetPassword() -

```
wdContext.nodeBapi_User_Change_Input().invalidate();

try {
	Bapi_User_Change_Input reset = new Bapi_User_Change_Input();

	reset.setUsername(
		wdContext.currentPasswordChangeElement().getUserName);

	Bapipwd pwd = new Bapipwd();
	//	A different password generator can also be used
	pwd.setBapipwd("Welcome123");
	reset.setPassword(pwd);

	Bapipwdx pwdx = new Bapipwdx();
	pwdx.setBapipwd(true);
	reset.setPasswordx(pwdx);

	wdContext.nodeBapi_User_Change_Input().bind(reset);

	wdContext
		.nodeBapi_User_Change_Input()
		.currentBapi_User_Change_InputElement()
		.modelObject()
		.execute();

	IPrivateResetpasswordView.IReturn_Reset_PasswordElement retMsg =
		wdContext
			.nodeReturn_Reset_Password()
			.currentReturn_Reset_PasswordElement();
	if (retMsg != null && retMsg.getType().equalsIgnoreCase("E")) {
		wdComponentAPI.getMessageManager().reportException(
			"Error Message - " + retMsg.getMessage(),
			false);
	} else if (
		retMsg != null && retMsg.getType().equalsIgnoreCase("S")) {
		wdComponentAPI.getMessageManager().reportSuccess(
			"Success - " + retMsg.getMessage());
	} else {
		wdComponentAPI.getMessageManager().reportWarning(
			"Warning - " + retMsg.getMessage());
	}

} catch (WDDynamicRFCExecuteException e) {
	wdComponentAPI.getMessageManager().reportException(
		"Exception - " + e.getMessage(),
		false);

	e.printStackTrace();
```

Related Content

Logon Screen Customization

Password Rules for SAP Backend System

How to use RFC Model in Webdynpro Java

For more information, visit the Webdynpro Java Homepage

Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.