```
<html><head><LINK REL=stylesheet HREF="/irj/portalapps/com.sap.portal.design.portaldesigndata/themes/portal/IBM_SAP/glbl/glbl_nn7.css?7.0.25.0
<LINK REL=stylesheet HREF="/irj/portalapps/com.sap.portal.design.portaldesigndata/themes/portal/IBM_SAP/prtl_std/prtl_std_nn7.css?7.0.25.0.2">

<!-- EPCF: BOB Core -->
<meta http-equiv="Content-Script-Type" content="text/javascript">
<script src="/irj/portalapps/com.sap.portal.epcf.loader/script/optimize/js13_epcf.js?7.00001691"></script>
<script>
<!--
EPCM.relaxDocumentDomain();
EPCM.init( {
Version:7.00001691,
Level:1,
DynamicTop:false, // [service=true nestedWinOnAlias=false]
UAType:21, // [Mozilla]
UAVersion:5.0,
UAPlatform:1, // [Win]
UIPMode:"2", // [Default=2, User=0, Personalize=true]
UIPWinFeatures:"",
UIPPortalPath:"https://magn54.pok.ibm.com:50501/irj/portal",
UIPPopupComp:"https://magn54.pok.ibm.com:50501/irj/servlet/prt/portal/prtroot/com.sap.portal.epcf.admin.WorkProtectPopup",
UIPPopupCompSize:"dialogWidth:450px; dialogHeight:200px; status:no",
UIPPopupMsgNN:"Your\x20current\x20page\x20contains\x20unsaved\x20data.\r\nDo\x20you\x20want\x20to\x20continue\x20with\x20navigation\x20and\x20
UIPPopupMsgND:"Your\x20current\x20page\x20contains\x20unsaved\x20data.\r\nDo\x20you\x20want\x20to\x20discard\x20the\x20changes\x20and\x20open\
DBGException:false
} );
EPCM.DSM.init( {
TerminatorURL:"/irj/servlet/prt/portal/prtroot/com.sap.portal.dsm.Terminator",
WinEmptyUrl:"/irj/portalapps/com.sap.portal.dsm/images/empty.gif",
NavAcrossSubFramesUrl:"disabled",
ForcedUserDebug:false,
KeepAliveActive:false,
KeepAliveDelta:840,
KeepAliveStopAfter:36000
} );
function SAPWP_receiveSessInfo( sessInfo, frameRef ){
  EPCM.DSM.processSession( sessInfo, frameRef );
}
//-->
</script>
<!-- EPCF: EOB Core -->

<script type="text/javascript">
/*HTML Business for Java, NW04S_25_REL, 131576, Mon Feb 13 20:16:26 GMT 2012*/
ur_system = {doc : window.document , mimepath :"/irj/portalapps/com.sap.portal.design.urdesigndata/themes/portal/IBM_SAP/common/", stylepath :
</script>
<title >SAP&#x20;NetWeaver&#x20;Portal</title><meta HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=UTF-8"><script SRC="/irj/portalapps/
<!-- EPCF: Component com.sap.portal.navigation.portallauncher.default, lhdnkjpngcedochkkfhgcjdgpdbhdejk -->

<!-- EPCF: Component com.sap.portal.pagebuilder.pageBuilder, oaaikhkjcdhbblknlpfgepdgpdbhdejl -->
<Script>
var caEngine = new CAManager('/irj/servlet/prt/portal/prtroot/com.sap.portal.httpconnectivity.httpauthentication.Engine','dialogHeight:10;dial
caEngine.registerCAEvent('com.sap.portal.httpconnectivity.httpauthentication','Teach',caEngine,'eventCallBack');
</Script>











<script>

var disableWorkProtectCheck = false;


function popupUnsavedDataBeforeUnload(evt)
{
        if ((typeof pageTitleBar != "undefined") && pageTitleBar.backForwardLink)
        {
                pageTitleBar.backForwardLink = false;
        }
        else
        {
                evt = (evt) ? evt : ((window.event) ? event : null);
                if ( EPCM.getUAType() != EPCM.MSIE && EPCM.getUAType() != EPCM.MOZILLA) return;
                if ( EPCM.getGlobalDirty() && (! disableWorkProtectCheck ))
                {
                        if(EPCM.getUAType() == EPCM.MSIE )
                        {
                                evt.returnValue = 'You have unsaved data';
                        }
                        else
                        {
                                evt.preventDefault();
                                evt.stopPropagation();
                                return 'You have unsaved data';
                        }
                }
        }
}

try{
     if ( EPCM.getUAType() == EPCM.MSIE || EPCM.getUAType()== EPCM.MOZILLA){
       if (window==EPCM.getSAPTop()){
         window.onbeforeunload = popupUnsavedDataBeforeUnload;
       }
     }
} catch(ex){}
</script><script>frameworkSupport.init({anonymous:false,phase:'framework',portalURL:'https://magn54.pok.ibm.com:50501/irj/portal'});</script><



<script>
        // The Locker object for enablinng multiple OBN requests without aborting the previous ones.
        // The form locker object introduces 2 methods that implement FIFO algorithm in very thin manner adapted to OBN needs:
        //      1. getOldestEventObjectCopy - that returns the oldest copy of the eventObject that contains
        //              the OBN data. The eventObject is removed from the eventObjQueue.
        //      2. insertNewEventObjectCopy - inserts an eventObject that contains an OBN data into eventObjQueue.
        //              The form locker object introduces a member for handling a multiple submits of this form properly:
```

```
        // The boolean property isFormLocked is initially 'false'. It is modified from 2 locations:
        //      1. Externally: from the NavigationFramework.js - onObjBasedNavigate method for the Classic FW (the property is set to
        //              'true' at the very begining of the method in case the form is not already locked)
        //      2. From the onIFrameLoad method which is called when the form load is complete ('onLoad' event of the 'obnNavIFrame')
        //              The isFormLocked is set to 'false' as soon as the 'obnNavIFrame' is loaded.
        var OBNFormLocker = function() {
                this.isFormLocked = false;
                this.eventObjQueue = [];
                // the actual size of the event objects queue - initialy zero
                this.queueActualSize = 0;

                this.getOldestEventObjectCopy  = function() {
                        var element = undefined;
                        if ( this.eventObjQueue.length ){
                                // get the oldest event object
                                element = this.eventObjQueue[this.queueActualSize];
                                // update the buffer and check whether the shift is needed
                                if ( ++this.queueActualSize * 2 >= this.eventObjQueue.length ){
                                        // truncate the buffer
                                        this.eventObjQueue = this.eventObjQueue.slice( this.queueActualSize );
                                        // reset the actual size of the event objects queue
                                        this.queueActualSize = 0;
                                }
                        }
                        // return the removed event object
                        return element;
                }

                this.insertNewEventObjectCopy = function( eventObjCopy ) {
                        this.eventObjQueue.push( eventObjCopy );
                }

        }
        // Instantiate the OBNFormLocker object
        var obnFormLocker = new OBNFormLocker();

        // Calling object based navigation method of the appropriate framework.
        // The oldest coopy of the eventObject that contains an OBN data retrieved from the
        // obnFormLocker object and passed to the object based navigation method as a parameter.
        var callOnObjNav = function() {
                var oldestEventObjCopy = obnFormLocker.getOldestEventObjectCopy();
                if( oldestEventObjCopy != "undefined" && oldestEventObjCopy != null ) {
                        onObjBasedNavigate( oldestEventObjCopy );
                }
        }
        // This method is triggered by the 'onLoad' event of the browser for the "obnNavIFrame".
        function onIFrameLoad(){
                obnFormLocker.isFormLocked = false;
                callOnObjNav();
        }
</script>
<span id=divChangeContent name=divChangeContent style="position:absolute;height:0;width:0;top:-5000;left:-5000">
        <FORM action="" method=POST id="frmChangeContent" name="frmChangeContent">
                <INPUT type="hidden" id=NavigationTarget name=NavigationTarget>
                <INPUT type="hidden" id=RelativeNavBase name=RelativeNavBase></INPUT>
        </FORM>
        <form  id='obnNavForm' method='post' target='obnNavIFrame' >
                <input type='hidden' name='systemAlias'>
                <input type='hidden' name='businessObjName'>
                <input type='hidden' name='objValue'>
                <input type='hidden' name='operation'>
                <input type='hidden' name='usePost' value='false'>
                <input type='hidden' name='source'>
                <input type='hidden' name='resolvingMode' value='Default'>
        </form>
</span>
<iframe src='/irj/portalapps/com.sap.portal.pagebuilder/html/EmptyDocument.html' style='position:absolute;height:0;visibility:hidden' name='ob
<script>var disablePersonalize = true;</script><SCRIPT>var emptyDocumentUrl = "/irj/portalapps/com.sap.portal.pagebuilder/html/EmptyDocument.h
<!-- EPCF: Component com.sap.portal.layouts.framework.framework, kjpmighjkkpoafkncbacgddgpdbhdmpl -->





<SCRIPT>if (typeof EPCM != "undefined") {EPCM.relaxDocumentDomain();} else { var d=document.domain; if (d.search(/^\d+\.\d+\.\d+\.\d+$/)>=0) {
pageSupport.pageHelperUrl = '/irj/servlet/prt/portal/prtroot/com.sap.portal.pagebuilder.PageHelper';
pageSupport.proxyModesUrl = '/irj/servlet/prt/portal/prtroot/com.sap.portal.pagebuilder.IviewModeProxy';
pageSupport.addPageId('pcd:portal_content/com.ibm.hrms.IBM_Layout-F/IBM-D/frameworkPages/com.ibm.hrms.IBM_frameworkpage-FP','0','local');
pageSupport._addIvuPageId("pcd\x3aportal_content\x2fcom.ibm.hrms.IBM_Layout\x2dF\x2fIBM\x2dD\x2fframeworkPages\x2fcom.ibm.hrms.IBM_frameworkpa
pageSupport._addIViewBank("page0ivu0",new iviewBank("","",pageSupport.EMBEDDED,1,"0","","GET","false"));
pageSupport._addIvuPageId("pcd\x3aportal_content\x2fcom.ibm.hrms.IBM_Layout\x2dF\x2fIBM\x2dD\x2fframeworkPages\x2fcom.ibm.hrms.IBM_frameworkpa
pageSupport._addIViewBank("page0ivu1",new iviewBank("","",pageSupport.EMBEDDED,1,"0","","GET","false"));
pageSupport._addIvuPageId("pcd\x3aportal_content\x2fcom.ibm.hrms.IBM_Layout\x2dF\x2fIBM\x2dD\x2fframeworkPages\x2fcom.ibm.hrms.IBM_frameworkpa
pageSupport._addIViewBank("page0ivu2",new iviewBank("","",pageSupport.EMBEDDED,1,"0","","GET","false"));
pageSupport._addIvuPageId("pcd\x3aportal_content\x2fcom.ibm.hrms.IBM_Layout\x2dF\x2fIBM\x2dD\x2fframeworkPages\x2fcom.ibm.hrms.IBM_frameworkpa
pageSupport._addIViewBank("page0ivu3",new iviewBank("","",pageSupport.URL,1,"0","","GET","false"));
</SCRIPT>
    <script>
                var scrollLocation=0;
        document.body.style.margin=0;
        document.body.scroll = "no";


    </script>


    <script defer language="JavaScript">ur_system.is508=false;ur_language="en";;</script><table cellspacing="0" cellpadding="0" width="100%" >
<!-- EPCF: Component common.hiddenjsp.HiddenJSP, flblclecakfdjjflkpjnibdgpdbhdekl -->








<!-- START IBM FSS R2.71 Test-->
<script type="text/javascript" language="javascript" src="/irj/portalapps/common.hiddenjsp/fss.jsp">
</script>

<!-- END IBM FSS -->
```

```
</TD></TR><TR><TD>
<!-- EPCF: Component com.sap.portal.navigation.toplevel.default, jnglekdglmbcmcfijpifhpdgpdbhdekc -->







<script>

var ZKey          =  90;
var zKey          = 122;
var BKey          =  66;
var bKey          =  98;
var leftKeyCode   =  37;
var rightKeyCode  =  39;
var upKeyCode     =  38;
var downKeyCode   =  40;

var levels                 = 2;
var isSection508SupportOn = false;
var isRtL                  = false;


function DoOnLoad() {
   if(gIsLoaded) {
           return;
   }
   gIsLoaded = true;

        printLevel1Table(gNavTree, -1, -1);
        printLevel2Table(gNavTree.children[0], -1, -1);


   gLevelOneActiveID=0;
   gLevelOneOldActiveID=0;
   SetTLNHeightAndSize();
   SetTLNHeightAndSize();//IE Bug?
   SetTopPosition();
   if(gIsPreviewMode) {
          PrintHoveringTLN();
   }

}

function PrintHoveringTLN() {
        gTLNNum = 2;
        var id = 1;
        gLevelOneHoverID = id;
        gHoverMode = true;


        printLevel1Table(gNavTree, gLevelOneActiveID, id);
        printLevel2Table(gNavTree.children[id], gLevelTwoActiveID, id);



        gHoverMode = false;
        SetTLNSize(true);
        SetTopPosition();
}

function ScrollTLN(side) {
   //If preview mode do nothing
   if (gIsPreviewMode) {
          return;
   }
   var offset        = side=='RIGHT'? 5 : -5;
   var divElement    = TLN_getElementById("TLNDiv");
   var divScrollLeft = divElement.scrollLeft;

   var levelOneEndElem = TLN_getElementById("LevelOneEnd");

   var levelTwoEndElem = TLN_getElementById("LevelTwoEnd");
   if(isRtL) {
          divElement.scrollLeft -= offset;
   } else {
          var leftPos = parseInt(divElement.style.left);
          if(isNaN(leftPos)) {
                 leftPos = 0;
          }
          leftPos -= offset;
          if(divScrollLeft>0) {
                 leftPos -= divScrollLeft;
                 divElement.style.width = parseInt(divElement.style.width) + divScrollLeft;
                 divElement.scrollLeft = 0;
          }
          // Exceeding minimum?
          if(leftPos > gLeftPos && offset < 0) {
                 return;
          }
          // Exceeding maximum?

          if(levelOneEndElem.offsetLeft + leftPos + gLeftPos < divElement.parentNode.offsetWidth &&
                 levelTwoEndElem.offsetLeft + leftPos + gLeftPos < divElement.parentNode.offsetWidth && offset > 0) {
                 return;
          }

          divElement.style.left  = leftPos;
          divElement.style.width = parseInt(divElement.style.width) + offset;
   }
}



var actionForLongTitles = "Allow";
var maxTitleLength      = 30;

function render(level, id, isActive, isHover, isLast, firstLevelStyle) {
        var tdClassName       = "prtlTopNav";
        var aClassName        = "prtlTopNav";
        var lastTD            = "";
        var sepTD             = ""
        var sepClassSuffix    = "";
        var onMouseClickEvent = " onclick=\"doMouseClick("+level+","+id+");return false;\" ";
        var onMouseEnterEvent = ""

        if(gHoverMode) {
                sepClassSuffix = "Hover";
        }
        if(level==1 || firstLevelStyle) {
                tdClassName += "1stLvl";
                if(isHoveringOn) {
                   onMouseEnterEvent = " onmouseover=\"doOnMouseEnter("+level+","+id+")\" ";
                }
        } else {
                tdClassName += "2ndLvl";
                aClassName  += "2nd";
                if(!isLast) {
                   sepTD =" <TD nowrap class=\"prtlTopNav2ndLvlSep"+sepClassSuffix+"\"> | </TD>"
                }
        }
        if(isActive) {
                tdClassName += "-a";
                aClassName  += "Act";
        } else {
                tdClassName += "-i";
                aClassName  += "Lnk";
        }
        if((level==1 || firstLevelStyle) && isHover && !isActive) {
                tdClassName = "prtlTopNav1stLvlHover";
                aClassName  = "prtlTopNavLnkHover";
        }
        if(gHoverMode && (level==2)) {
                tdClassName = "prtlTopNav2ndLvl-iHover";
                aClassName  = "prtlTopNav2ndLnk";
        }

        var title508     = "";
        var nodeName     = "";
        var accessKey    = "";
        var tabIndex     = "-1";
        var visibleTitle = this.title;
        if(actionForLongTitles=="Truncate" && this.title.length>maxTitleLength) {
                visibleTitle = this.title.substring(0,maxTitleLength) + "...";
                // In case of an accessible user, this title will be overriden
                title508 = " title=\"" + this.title + "\"";
        }
        //allow stopping at the TLN with tab also when not in ACC mode
        if (id==0){
                if (level==1){
                        tabIndex = "0";
                }
        }

   return "<TD nowrap id=\"navNode_" + level + "_" + id + "\" " +
                  "onkeydown=\"navNodeKeyDownHandler(" + level + ", " + id + ")\" " +
                  "class=\"" + tdClassName + "\" " + onMouseClickEvent + onMouseEnterEvent + ">" +
                  "<A id=\"navNodeAnchor_" + level + "_" + id + "\" " + title508 + " href=\"#\" " + nodeName + " class=\"" + aClassName + "\" 
                  "</TD>" + sepTD + lastTD;
}

function navNodeKeyDownHandler(level, position) {
   var keyCode = window.event.keyCode;
   switch(window.event.keyCode) {
          case leftKeyCode:  if(isRtL) {
                                                        moveFocusToNavNode(level, position, "right"); break;
                                                } else {
                                                        moveFocusToNavNode(level, position, "left"); break;
                                                }
          case rightKeyCode: if(isRtL) {
                                                        moveFocusToNavNode(level, position, "left"); break;
                                                } else {
                                                        moveFocusToNavNode(level, position, "right"); break;
                                                }
          case upKeyCode:    moveFocusToNavNode(level, position, "up"   ); break;
          case downKeyCode:  moveFocusToNavNode(level, position, "down" ); break;
          default: if((keyCode==ZKey || keyCode==zKey) && window.event.altKey && window.event.shiftKey) {
                                setFocusToFirstOrLastNav(true);
                        } else if((keyCode==ZKey || keyCode==zKey) && window.event.altKey ) {
                                setFocusToFirstOrLastNav(false);
                        }; break;
   }
   return true;
}

var numOfNodes         = new Array();
currentLevelOneNavNode = 0;

function moveFocusToNavNode(level, position, direction) {
   var numberOfNodes = 0;
   if(levels==1) {
          numberOfNodes = gNavTree.children.length;
   } else if(level==1) {
          numberOfNodes = gNavTree.children.length;
   } else {
          numberOfNodes = gNavTree.children[currentLevelOneNavNode].children.length;
   }
   if(direction=="left") {
          if(position==0) {
                TLN_getElementById("navNodeAnchor_" + level + "_" + (numberOfNodes-1)).focus();
          } else {
                TLN_getElementById("navNodeAnchor_" + level + "_" + (position-1)).focus();
          }
   } else if(direction=="right") {
          if(position==(numberOfNodes-1)) {
                TLN_getElementById("navNodeAnchor_" + level + "_0").focus();
          } else {
                TLN_getElementById("navNodeAnchor_" + level + "_" + (position+1)).focus();
          }
   } else if(direction=="up" && level>1) {
          TLN_getElementById("navNodeAnchor_" + (level-1) + "_0").focus();
   } else if(direction=="down") {
          if(level<levels) {
                TLN_getElementById("navNodeAnchor_" + (level+1) + "_0").focus();
          } else { // Goto the UIService's topmost node
                 // Option: To DTN...
          }
   }
}

function setFocusToFirstOrLastNav(focusOnFirst)
{
        var id = focusOnFirst ? "TLNTable" : "TLNLeaving"
        document.getElementById(id).focus();
/*   var name = focusOnFirst ? "firstTLNNavNode" : "lastTLNNavNode";
   document.getElementsByName(name)[0].focus();*/
}



var activeOneId = -1;
function keepTLNFocus(level,id)
{

        var id = (level==1)? "navNode_1_" + id : "navNode_2_" + id;

        var tdNode = TLN_getElementById(id);
        if(tdNode!="undefined"  && tdNode!=null)
                tdNode.children[0].focus();
}

function doMouseClick(level, id) {
        var cancelled = false;
        var activeOneId_OrigValue = activeOneId;
        var gLevelOneOldActiveID_OrigValue = gLevelOneOldActiveID;
        var gLevelOneActiveID_OrigValue = gLevelOneActiveID;
        //If preview mode do nothing
        if (gIsPreviewMode)
                return;
  // For ACC users the focus should stay on the TLN
        if(isSection508SupportOn)
                window.setTimeout("keepTLNFocus(" + level +"," +id+ ")",1500);
        if (level == 1)
        {
                curNode = gNavTree.children[id];
                if (!EPCM.getGlobalDirty())
                {
                        printLevel1Table(gNavTree, id);
                        printLevel2Table(curNode, -1, -1, true);
                }
                else
                {
                        activeOneId = gLevelOneActiveID;

                        //saving the old active id, in case we need to still keep it:
                        //When navigating in the TLN when work protect is enabled and its configured to open a navigation
                        //in a new window, then the active TLN first level should be the old one, and not the updated new value.
                        gLevelOneOldActiveID =  gLevelOneActiveID;
                        gLevelOneActiveID = id;
                }
                if (curNode.showType == 1)
                {
                        if (curNode.windowFeatures == "")
                          winFeatures = "height=" + curNode.windowHeight + ", width=" + curNode.windowWidth + ",toolbar,location,status,scroll
                        else
                        {
                                winFeatures = curNode.windowFeatures;
                                if (curNode.windowFeatures.indexOf("height") == -1)
                                        winFeatures = "height=" + curNode.windowHeight + ", "+winFeatures;
                                if (curNode.windowFeatures.indexOf("width") == -1)
                                        winFeatures = "width=" + curNode.windowWidth + ", "+winFeatures;
                        }
                        //we must add ExecuteLocally=true because the navigation came from TLN
                        //(if the node name does not contain it already).
                        if(curNode.name.indexOf("ExecuteLocally") < 0) {
                                if(curNode.name.indexOf("?") > 0) {
                                        curNode.name += "&ExecuteLocally=true";
                                }
                                else {
                                        curNode.name += "?ExecuteLocally=true";
                                }
                        }
                        cancelled = EPCM.doNavigate(curNode.name,1, winFeatures, curNode.windowName);
                }
                else
                {

                                var additionalParams = "InitialNodeFirstLevel=true";

                        //check if the node has paremeters already.
                        if(curNode.name.indexOf("?") > 0) {
                                additionalParams = "&" + additionalParams;
                        }
                        else {
                                additionalParams = "?" + additionalParams;
                        }
                        cancelled = EPCM.doNavigate(curNode.name+additionalParams);
                }
                //if first level in the TLN was pressed and in on unsaved data popup cancel button choosed,
                //than we need to revert all active id's calculations.
                if(cancelled) {
                        activeOneId = activeOneId_OrigValue;
                        gLevelOneOldActiveID = gLevelOneOldActiveID_OrigValue;
                        gLevelOneActiveID = gLevelOneActiveID_OrigValue;
                }
        }
        else
        {
                if (gLevelOneHoverID != -1)
                        gLevelOneActiveID = gLevelOneHoverID;
                activeOneId = gLevelOneActiveID;

                                curNode = gNavTree.children[gLevelOneActiveID].children[id];


                if ((curNode.showType == 0) && !EPCM.getGlobalDirty()) {

                                printLevel1Table(gNavTree, gLevelOneActiveID, gLevelOneHoverID);
                                printLevel2Table(gNavTree.children[gLevelOneActiveID], id, gLevelTwoHoverID);

                        var additionalParams = "NavPathUpdate=false";
                        //check if the node has paremeters already.
                        if(curNode.name.indexOf("?") > 0) {
                                additionalParams = "&" + additionalParams;
                        }
                        else {
                                additionalParams = "?" + additionalParams;
                        }
                        EPCM.doNavigate(curNode.name + additionalParams);
                }
                else {
                        if (curNode.showType == 0) {
                                EPCM.doNavigate(curNode.name);
                        }
                        else {
                                if (curNode.windowFeatures == "")
                                winFeatures = "height=" + curNode.windowHeight + ", width=" + curNode.windowWidth + ",toolbar,location,status,
                                else {
                                        winFeatures = curNode.windowFeatures;
                                        if (curNode.windowFeatures.indexOf("height") == -1)
                                                winFeatures = "height=" + curNode.windowHeight + ", "+winFeatures;
                                        if (curNode.windowFeatures.indexOf("width") == -1)
                                                winFeatures = "width=" + curNode.windowWidth + ", "+winFeatures;
                                }
                                //we must add ExecuteLocally=true because the navigation came from TLN
                                //(if the node does not contain it already).
                                if(curNode.name.indexOf("ExecuteLocally") < 0) {
                                        if(curNode.name.indexOf("?") > 0) {
                                                curNode.name += "&ExecuteLocally=true";
                                        }
                                        else {
                                                curNode.name += "?ExecuteLocally=true";
                                        }
                                }
                                EPCM.doNavigate(curNode.name, curNode.showType, winFeatures,curNode.windowName);
                        }
                }
                if (activeOneId != -1) {
                        //if activeOneId was changed then update gLevelOneActiveID
                        gLevelOneActiveID = activeOneId;
                        activeOneId = -1;
                }
        }
}


function onUpdateTLN(eventObj) {
   //Find the nodeid in the navigation tree
   for(i=0;i<gNavTree.children.length;i++) {
         if(gNavTree.children[i].name==eventObj.dataObject) {



                 printLevel1Table(gNavTree, i);
                 printLevel2Table(gNavTree.children[i], -1);



                 currentLevelOneNavNode = i;
                 SetTLNSize(true);
                 adjustFocusToNode(TLN_getElementById("navNode_1_" + i).firstChild);
                 return;
         } else {
                 for(j=0;j<gNavTree.children[i].children.length; j++) {
                        if(gNavTree.children[i].children[j].name==eventObj.dataObject) {



                           printLevel1Table(gNavTree, i);
                           printLevel2Table(gNavTree.children[i], j);



                           SetTLNSize(true);
                           adjustFocusToNode(TLN_getElementById("navNode_2_" + j).firstChild);
                           return;
                        }
                 }
         }
   }
}

function onUpdateTLNByBrowser(eventObj) {
   //Find the nodeid in the navigation tree
   for(i=0;i<gNavTree.children.length;i++) {
         for(j=0;j<gNavTree.children[i].children.length; j++) {
                 var name = gNavTree.children[i].children[j].name + "/";
                 if(eventObj.dataObject==gNavTree.children[i].children[j].name ||
                        eventObj.dataObject.indexOf(name)!= -1) {



                        printLevel1Table(gNavTree, i);
                        printLevel2Table(gNavTree.children[i], j);


                        SetTLNSize(true);
                        adjustFocusToNode(TLN_getElementById("navNode_2_" + j).firstChild);
                        return;
                 }
         }
         if(eventObj.dataObject.indexOf(gNavTree.children[i].name)!=-1) {



                 printLevel1Table(gNavTree, i);
                 printLevel2Table(gNavTree.children[i], 0);



                 currentLevelOneNavNode = i;
                 SetTLNSize(true);
                 adjustFocusToNode(TLN_getElementById("navNode_1_" + i).firstChild);
                 return;
          }
   }
}



<!---------------------->
function onUpdateTLN2(eventObj)
{

   var index1st;
   var index2nd;
   var navArray;

   //get the last path indexes for the context
   var entry = EPCM.getSAPTop().gHistoryFrameworkObj.GetLastEntry();
   if (entry==null)
   {
        navArray = new Array();
    navArray[0] = 0;
    navArray[1] = 0;
   }
   else
   {
        navArray = entry.getPathIndexes();
   }


   var pathLength = navArray.length;
   if (pathLength <1){
     return;
   }
   index1st = navArray[0];
   if(isHoveringOn) {
           gLevelOneHoverID = index1st;
   }
   if(pathLength>=2){
      index2nd = navArray[1];
   }
   currentLevelOneNavNode = index1st;
   if(index2nd==null || index2nd=="undefined")
     index2nd = -1;
   if(pathLength==1)
   {

                printLevel1Table(gNavTree, index1st);
                printLevel2Table(gNavTree.children[index1st],index2nd);


                SetTLNSize(true);

                        adjustFocusToNode(TLN_getElementById("navNode_1_" + index1st).firstChild);

                return;
        }

           printLevel1Table(gNavTree, index1st);
           printLevel2Table(gNavTree.children[index1st], index2nd);

                   SetTLNSize(true);

                   adjustFocusToNode(TLN_getElementById("navNode_2_" + index2nd).firstChild);

                   return;
}

function onUpdateTLNByBrowser2(eventObj)
{


   var index1st;
   var index2nd;
   var navArray;

   //get the last path indexes for the context
   var entry = EPCM.getSAPTop().gHistoryFrameworkObj.GetLastEntry();
   if (entry==null)
   {
        navArray = new Array();
    navArray[0] = 0;
    navArray[1] = 0;
   }
   else
   {
        navArray = entry.getPathIndexes();
   }

   var pathLength = navArray.length;
   if (pathLength <1){
     return;
   }
   index1st = navArray[0];
   if(isHoveringOn) {
        gLevelOneHoverID = index1st;
   }
   if(pathLength>=2){
      index2nd = navArray[1];
   }
   if(index2nd==null || index2nd=="undefined")
     index2nd = -1;


                printLevel1Table(gNavTree, index1st);
                printLevel2Table(gNavTree.children[index1st],index2nd);

        if(pathLength>=2)
        {
                SetTLNSize(true);

                adjustFocusToNode(TLN_getElementById("navNode_2_" + index2nd).firstChild);

                return;
        }
        currentLevelOneNavNode = index1st;
        SetTLNSize(true);

        adjustFocusToNode(TLN_getElementById("navNode_1_" + index1st).firstChild);


        return;
}
<!------------------------->

EPCM.subscribeEvent("urn:com.sapportals:navigation", "UpdateTLNByBrowser", onUpdateTLNByBrowser2);
EPCM.subscribeEvent("urn:com.sapportals:navigation", "UpdateTLN"         , onUpdateTLN2         );



EPCM.subscribeEvent("urn:com.sapportals.portal:browser"    , "load"  , DoOnLoad);
EPCM.subscribeEvent("urn:com.sapportals:toplevelnavigation", "Onload", DoOnLoad);

</script>
<script>isHoveringOn=false;gNavTree = new NavNode("Top", "Top", 0, 0, 0, 0, '',new NavNode("navurl\x3a\x2f\x2f24dffaa280f9ec7c237ce7456872c87a







<table id="TLNTable" name="TLNTable" border="0" onresize="SetTLNHeightAndSize()" cellspacing="0" cellpadding="0" class="prtlTopNavWhl" tabInde
  <tr>
        <td id="NotchTD" name="NotchTD" nowrap class="prtlTopNavNotch" style="{position:absolute;}">&nbsp;</TD>
        <td>
          <div width="100%">&nbsp;</div>
          <div id="TLNDiv" name="TLNDiv" class="prtlTopNavContainer" onScroll="localScrollLeft = this.scrollLeft; adjustLeftAndWidth(this); re
                <!-- 1st level start -->
                <div id="Level1DIV">
                  <TABLE id="level1" name="level1" border="0" cellspacing="0" cellpadding="0">
                  </TABLE>
                </div>
                <!-- 2nd level start -->
                <div id="Level2DIV" class="prtlTopNav2ndLvlWhl">
                  <TABLE id="level2" name="level2" border="0" cellspacing="0" cellpadding="0" height="100%">
                  </TABLE>
                </div>

          </div>
        </td>
        <td nowrap id="LeftScroll"  name="LeftScroll"   class="prtlDivScrollerLeft"   style="cursor:hand" onmousedown="StartScrollTLN('LEFT')"
        <td nowrap id="RightScroll" name="RightScroll"  class="prtlDivScrollerRight"  style="cursor:hand" onmousedown="StartScrollTLN('RIGHT')
        <td nowrap id="HeightSpacer" name="HeightSpacer" style="display:none">&nbsp;</td>
  </tr>
</table>

<!-- Contnet Separator -->
<tr>
                <td class="prtlTopNavContentSep">
                </td>
        </tr>
<!-- end of content sparator -->

<SCRIPT>
if(isIE) {
   var table = document.getElementById('TLNTable');
   var myParent = table.parentNode;
   var highestTable = table;
   while(myParent.tagName!="BODY") {
          if(myParent.tagName=="TABLE") {
                 highestTable = myParent;
          }
          myParent = myParent.parentNode;
   }
   highestTable.onresize = SetTopPosition;
}
</SCRIPT>



<script>
gNumOfTLNs++;
</script>
</TD></TR><TR><TD>
<!-- EPCF: Component com.sap.portal.navigation.pagetoolbar.PageToolbar, ngobpjcheeaaddflohneondgpdbhdekh -->
<script defer language="JavaScript">ur_system.is508=false;ur_language="en";;</script><table cellspacing="0" cellpadding="0" width="100%" ><tr>
pageTitleBar.Mode = 'default';pageTitleBar.breadId='BreadCrumbDiv';pageTitleBar.acc=false;pageTitleBar.showBack = true;pageTitleBar.HistoryMen
<script>
function setTitle(){
if (EPCM.getSAPTop().gHistoryFrameworkObj.GetActiveTrackingEntryValue())
 var title=EPCM.getSAPTop().gHistoryFrameworkObj.GetActiveTrackingEntryValue().title;
if (title != null){
document.getElementById("innerPageWrapper").tt=title;
}
}
window.setTimeout("setTitle()",3000)
</script></TD></TR></TABLE>
        </td></tr><tr><td></td></tr></table>

<SCRIPT>
     pageSupport.adjustFullPageIViews();
     if (typeof EPCM != "undefined") { EPCM.subscribeEvent( "urn:com.sapportals.portal:browser", "resize", pageSupport.adjustFullPageIViews);
     EPCM.subscribeEvent( "urn:com.sapportals.portal:browser", "load", pageSupport.adjustFullPageIViews); }
</SCRIPT>



    <script>
        // Set the Top Level Navigation iView size and position.
        EPCM.raiseEvent("urn:com.sapportals:toplevelnavigation", "Onload", null);
    </script>
 </body></html>
```