

# Versteckte User-Exits

## Workflow

Einige Skripte enthalten User-Exit Funktionen und viele dieser Funktionen sind in den verschiedenen "UserExit"-Bibliotheken enthalten. Wenn neue Funktionen hinzukommen müssten diese Bibliotheken allerdings immer manuell um diese Funktionen erweitert werden. Daher werden neue hinzugefügte UserExit Funktionen nur aufgerufen, wenn Sie als Funktion existieren. Die Funktionen können bei Bedarf in einer UserExit-Bibliothek (zum Beispiel "DEXPRO\_\_UserExit\_CustomLib") hinzugefügt werden.

Die Liste bezieht sich jeweils auf die aktuellste Version. Ggf. sind Funktionen in älteren Versionen nicht verfügbar.

### Bei der Initialisierung

```
/** Für initiale projektspezifische Ausführungen.  
 * Die Modul-spezifischen Funktionen werden nur beim spezifischen Mappentypen aufgerufen,  
wenn die Funktionen existieren.  
 * "ue_Initialization()" wird nachfolgend aufgerufen.  
 * @portalscript: DEXPRO_WF_Initialization  
 * @param {Log} log Logging-Objekt  
 * @returns {boolean} true/false  
**/  
function ue_Initialization(log){  
    ;  
}  
function ue_Initialization_Invoice(log){  
    ;  
}  
function ue_Initialization_Mailroom(log){  
    ;  
}  
function ue_Initialization_Procurement(log){  
    ;  
}
```

```

/** Für initiale projektspezifische Anpassungen am Gentable (nur Invoice).
 * @portalscript: DEXPRO_WF_Initialization
 * @param {Log} log Logging-Objekt
 * @returns {boolean} true/false
 */
Gentable.prototype.ue_Initialization_Invoice = function(log){
    ;
}

/** Wird bei der Erstellung einer initialen Gentable-Zeile ausgeführt.
 * @portalscript: DEXPRO_WF_Initialization
 * @param {string} optExecType Optionale Ausführungs-Info
("Initialization"/"ReplacePositionsByOrder")
 * @returns Kein Rückgabewert!
 */
Gentable.prototype.createInitialRow = function(optExecType){
    []
}

/** Direkt zu Beginn des Workflows kann über den Parameter "Workflow_InitialDelay"
 * eine Workflow-Verzögerung angesteuert werden.
 * Nur wenn die Verzögerung angesteuert wird, dann wird vorab dieses User-Exit ausgeführt.
 * Die Funktion wird lediglich ausgeführt.
 * @portalscript: DEXPRO_WF_CheckInitDelay
 * @returns Kein Rückgabewert!
 */
function ue_BeforeInitialDelay(){
    ;
}

/** Zu Beginn des Workflows wird das initiale Skript ausgeführt und im Anschluss werden die
Workflow-Regeln ermittelt.
 * In speziellen Fällen wie bei der Erstellung von Split-Mappen kann es gewünscht sein, dass
diese Aktionen übersprungen werden sollen.
 * @portalscript: DEXPRO_WF_InitAbbreviation
 * @param {Log} log Logging-Objekt
 * @returns {boolean} true (Abkürzung nehmen) / false
 */
function ue_InitAbbreviation(log){
    log.info("[ "+context.file.getid()+" ] start function ue_InitAbbreviation()");
    []return true;
}

```

```

}

function ue_InitAbbreviation_Invoice(log){
    return true;
}

function ue_InitAbbreviation_Mailroom(log){
    return true;
}

function ue_InitAbbreviation_Procurement(log){
    return true;
}

```

## Nach der Ermittlung der Workflow-Regeln

```

/** Die User-Exit-Funktion wird ausgeführt, wenn eine Aktion über die Workflow-Regeln
übersprungen wird.
 * @portalscript: DEXPRO_WF_Rules_ActionCheck_SkipAction
 * @returns {boolean} true / false (führt zu einem Workflow-Fehler)
**/
function ue_OnSkipAction(){
    return true;
}

/** Nach den Workflow-Regeln wird zunächst auf eine technische Aktion und auf das
Überspringen der Aktion geprüft.
 * Nur wenn die Aktion einer Gruppe oder einem Benutzer zugeordnet wird, wird das Skript
"DEXPRO_WF_Rules_BeforeUserAction" aufgerufen.
 * In dem Skript werden vor allem Zugriffsberechtigungen für die ermittelte Gruppe bzw. für
den ermittelten Benutzer gesetzt.
 * Über die Workflow-Regeln wurde bereits der Benutzer bzw. die Gruppe zugeordnet.
 * Bei einem Split wird das Skript für jede einzelne Split-Mappe ausgeführt!
 * @portalscript: DEXPRO_WF_Rules_BeforeUserAction
 * @returns {boolean} true / false (führt zu einem Workflow-Fehler)
**/
function ue_BeforeUserAction(){
    return true;
}

```

## Bei der Benutzer-Aktion

```
/** Das UserExit wird bei Übernahme einer Mappe aus einem Gruppenkorb aufgerufen.  
 * Der Aufruf erfolgt über das Skript "DEXPRO_WF_CheckDataForwardUser".  
 * In dem Skript wird zuvor lediglich der aktuelle Benutzer in das Feld "ActionUser"  
geschrieben.  
 * Bei einem Fehler muss die Fehlermeldung zurückgegeben werden - andernfalls ein Leerstring!  
 * @portalscript: DEXPRO_WF_CheckDataForwardUser  
 * @return {string} "" / Fehlermeldung.  
**/  
function ue_CheckDataForwardUser(){  
    return "";  
}  
  
/** Das UserExit wird beim Zurücklegen in den Gruppenkorb aufgerufen.  
 * Der Aufruf erfolgt über das Skript "DEXPRO_WF_CheckDataBackAccessProfile".  
 * Bei einem Fehler muss die Fehlermeldung zurückgegeben werden - andernfalls ein Leerstring!  
 * @portalscript: DEXPRO_WF_CheckDataBackAccessProfile  
 * @return {string} "" / error-message.  
**/  
function ue_CheckDataBackAccessProfile(){  
    return "";  
}
```

## Am Aktions-Ende

```
/** Die Funktionen werden bei Abschluss der Aktion zu Beginn bzw. am Ende ausgeführt.  
 * @portalscript: DEXPRO_WF_CheckActionEnd  
 * @returns {string} Leerstring oder Fehlermeldung  
**/  
function ue_OnActionEnd_Start(){  
    return "";  
}  
function ue_OnActionEnd_End(){  
    return "";  
}
```

```

/** Wird am Ende einer Workflow-Aktion ausgeführt.
 * Hierdurch muss das Gentable nicht mehrfach ausgelesen und geschrieben werden.
 * Das Gentable wird im Anschluss im aufrufenden Skript in das Gentable-Feld zurück
geschrieben.
 * @portalscript: DEXPRO_WF_CheckActionEnd
 * @returns Kein Rückgabewert benötigt. Fehler werden über this.Result und this.Error
gesteuert.
**/
Gentable.prototype.ue_OnActionEnd = function(optExecType){
    this.Error = "";
    this.Result = true;
}

/** Nach Abschluss einer Aktion kann eine Verzögerung angesteuert werden.
 * Dadurch kann der Anwender ggf. schneller weiter arbeiten.
 * @portalscript: DEXPRO_WF_CheckDelayAfterAction
 * @returns {boolean} true/false
**/
function ue_DelayAtWorkflowActionEnd(){
    return true;
}

```

## Am Workflow-Ende

```

/** Am Workflow-Ende sollte jeder Vorgang archiviert sein und Rechnungen sollten zum Beispiel
den Status "gebucht" haben.
 * Diese User-Exit Funktion wird standardmäßig in der "DEXPRO__UserExit_WorkflowLib"
ausgeliefert.
 * Die Prüfungen können beliebig angepasst werden.
 * @portalscript: DEXPRO_WF_CheckDataAtTheEndOfTheWorkflow
 * @return {boolean} true / false.
**/
function ue_CheckFileDataAtTheEndOfTheWorkflow(){
    return true;
}

/** Am Workflow-Ende kann eine Mappe gelöscht ("delete") oder versiegelt ("seal") werden oder

```

es passiert nichts mit dem Beleg.

- \* Die Information wird in der WEB-Konfiguration zur Workflow ID angegeben und kann über dieses UserExit pro Mappe manipuliert werden.

```
* @portalscript: DEXPRO_WF_End_Seal / DEXPRO_WF_End_Delete
* @param {string} wfEndType Ermittelte Angabe über die WEB-Konfiguration.
* @return {string} "delete" / "seal" / "".
**/

function ue_ManipulateWorkflowEndType( wfEndType){
    return wfEndType;
}
```

## Beim Zurücksenden

```
/** Das UserExit wird beim Zurücksenden über das Skript "DEXPRO_WF_CheckSendBack" aufgerufen.
* In neueren Versionen wurden UserExit-Funktionen pro Modul ergänzt.
* portalscript: DEXPRO_Action_SendBack / DEXPRO_Action_SendBack_EventReport /
Procurement_Action_SendBackRequester
* @return {boolean} true / false (Fehler).
**/

function ue_BeforeSendBack(){
    // context.errorMessage = "Fehlermeldung";
    return true;
}

/** Nur: DEXPRO_Action_SendBack_EventReport **/
function ue_BeforeSendBack_Invoice(){ return true; }
function ue_BeforeSendBack_Mailroom(){ return true; }
function ue_BeforeSendBack_Procurement(){ return true; }

/** Das UserExit wird beim Zurücksenden über das Skript "DEXPRO_WF_CheckSendBack" aufgerufen.
* @portalscript: DEXPRO_WF_CheckSendBack
* @return {string} Leerstring/ Fehlermeldung.
**/

function ue_OnSendBack(){
    return "";
}
```