

SAP Manufacturing Execution Mobile Developers Guide



Applicable Release: SAP ME 15.1

Version 1.0

October 2015

SAP ME 15.1 Mobile Developers Guide

© Copyright 2015 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 AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors. National product specifications may vary. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group 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. SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries. Please see <http://www.sap.com/corporate-en/legal/copyright/index.epx#trademark> for additional trademark information and notices.

Disclaimer

Please see <http://www.sap.com/corporate-en/legal/copyright/index.epx> for disclaimer information and notices.

Documentation in the SAP Service Marketplace

You can find this document at the following address: <http://service.sap.com/instguides>

Document History

<u>Document Version</u>	<u>Description</u>
1.0	Initial version

Table of Contents

1	Guide Overview	1
2	Introducing SAP ME Mobile	1
2.1	Users.....	1
2.1.1	User Prerequisites	1
2.2	Terms and Acronyms.....	2
3	SAP ME Mobile Application.....	2
3.1	SAP ME Mobile Activities.....	2
3.1.1	Model.....	3
3.1.2	View	3
3.1.3	Controller.....	3
3.1.4	Example View and Controller.....	3
3.1.4.1	Example View	3
3.1.4.2	Example Controller	4
3.1.5	View Navigation	5
3.1.6	Data Binding and OData Services.....	7
3.1.6.1	Retrieving data from Server	7
3.1.6.2	Sending transaction request to Server	8
3.1.7	Localized Text.....	8
3.1.8	Extended Text Controls.....	8
3.1.8.1	Input Control	9
3.1.8.2	TextArea Control	9
3.2	SAP ME Common Utility Functions	10
3.2.1	util.IOUtil Functions.....	10
3.2.2	util.I18NUtility Functions	12
3.2.3	util.Message Functions	13
3.2.4	util.Dialog Functions	13
3.2.5	util.Activity Functions	14
3.2.6	util.SystemRules Functions	15
3.2.7	util.Model Functions.....	15

SAP ME 15.1 Mobile Developers Guide

3.2.8	util.StringUtil Functions	17
3.2.9	util.CommonKey Constants.....	18
3.3	SAP ME Mobile Utility Functions	18
3.3.1	util.Common Functions.....	19
3.3.2	util.Navigation Functions.....	19
3.3.3	util.ModelKey Constants.....	19
4	SAP ME Mobile “Lean”	22
4.1	Application Example.....	26
4.1.1	Control Creation.....	26
4.1.2	Toolbar Creation.....	27
4.1.3	Implement Controls.....	27
5	SAP ME Mobile “Ultra-Lean”	32
5.1	Ultra-Lean JavaScript functions	43

1 Guide Overview

This developer's guide provides information required for developing Mobile applications for SAP ME. The following topics will be covered by this document:

Major Section	Description
Introducing SAP ME Mobile	A general introduction to SAP ME Mobile, including usage and term definitions.
SAP ME Mobile Application	Describes the SAP ME Mobile Application framework and how to develop Mobile activities.
SAP ME Mobile "Lean"	Describes an example "lean" version of a mobile application for use on devices that do not support HTML5 controls. This example is dependent on the SAPUI5 lean and JQuery JavaScript libraries.
SAP ME Mobile "Ultra-Lean"	Describes an example "ultra-lean" version of a mobile application dependent only on the "datajs.js" JavaScript library.

2 Introducing SAP ME Mobile

SAP ME Mobile is a SAPUI5 based application that allows users to perform SAP ME transactions on manufacturing mobile devices and supported platforms. The SAP ME Mobile application is a type of Production Operator Dashboard (POD) and is configured in POD Maintenance. SAP ME Mobile activities are assigned to the POD.

An SAP ME Mobile activity consists of a reference to a UI5 "view" to be displayed in the SAP ME Mobile application. Activities are configured in Activity Maintenance. For more information on configuring SAP ME Mobile, see the SAP Mobile How-To Guide. This document describes how to develop the view's to be assigned as activities.

SAP ME Mobile based applications must run in browsers that support HTML5. Many older mobile devices do not provide browsers capable of running SAP ME Mobile. This does not preclude the use of many of the underlying utilities that SAP ME Mobile and SAPUI5 provides. It is possible to develop a non-HTML5 application that will run in older browsers using older versions of HTML. While more difficult to develop due to the lack of the "rich" controls provided by SAPUI5, it is still possible to build simplified applications. A sample "lean" application is provided with SAP ME Mobile that describes how to implement an application that will run in older browsers. This is described in the section on [SAP ME Mobile "Lean"](#).

2.1 Users

SAP consultants or other System Integrators (programmers) that develop SAP ME Mobile activities to operate from the SAP ME Mobile POD.

2.1.1 User Prerequisites

In order to write SAP ME Mobile activities, the Integrator must have a working knowledge of JavaScript, HTML, SAPUI5, JQuery and OData Services.

2.2 Terms and Acronyms

Understanding the following terms is necessary to understand the ME Mobile application framework.

HTML

HyperText Markup Language is the standard language used to create web pages. UI5 controls render HTML5 markup.

JavaScript

JavaScript is the programming language for UI5.

JQuery

[jQuery](#) JavaScript library is used internally by UI5 as well as by SAP ME Mobile activities.

SAPUI5 / OpenUI5

A JavaScript UI library of UI controls that support data binding for different data sources. OpenUI5 is an open source “free” version of UI5. SAPUI5 is a version that may only be used by SAP customers. SAP ME Mobile utilizes the SAPUI5 library installed with the SAP NetWeaver server.

DATAJS

[Datajs](#) is a cross-browser JavaScript library that supports modern protocols such as JSON and OData. It is used by the ME Mobile Ultra-Lean example.

OData

A Java Library that implements the Open Data Protocol. SAP ME Mobile uses the libraries provided by the [Apache Olingo Project](#). SAP ME Mobile activities utilize OData services for accessing SAPME Public API's. See the SAP ME OData Services Guide for the services available to be used by ME Mobile activities.

POD

The Production Operator Dashboard provides the main view which displays and manages the interaction of mobile activities for the user. A POD is configured using SAP ME *POD Maintenance* activity.

3 SAP ME Mobile Application

The SAP ME Mobile application is used to render a list of SAP ME Mobile activities on a device's browser that is capable of rendering HTML5 content. The application is configured in POD maintenance using the “Mobile” POD type. This section describes how to develop activities for use in the application.

3.1 SAP ME Mobile Activities

SAP ME Mobile Activities follow the Model View Controller (MVC) concept as implemented by SAPUI5. The “view” is responsible for the definition and rendering of the UI, the “model” manages the application data and the “controller” reacts to user interactions and view events by modifying the

view and model. ME Mobile activities have a 1:1 relationship between the view and controller. This section provides a detailed description on developing mobile activities for the SAP ME Mobile application.

3.1.1 Model

The model holds the data and provides methods for setting and updating data. SAPUI5 provides 3 types of models; JSON Model, OData Model and an XML Model. More detailed information on these models can be found in the SAPUI5 documentation. ME Mobile core activities primarily use the JSON Model which is a client-side model that can be bound to UI5 controls for display.

3.1.2 View

The view defines the user interface. SAP ME Mobile activities are defined using a view defined using JavaScript (JS View).

3.1.3 Controller

The controller is responsible for reacting to user interactions and view events. It is responsible for modifying the view and model. The controller is a JavaScript file that is referenced by the view and provides the name of the “controller” that manages it.

3.1.4 Example View and Controller

The following example activity shows how an SAP ME Mobile activity should be defined. The following view is defined in a JavaScript file named `Example.view.js`. The controller that follows it is defined in a file named `Example.controller.js`. They are both defined in the package `com.sap.me.example.view` which would relate to `/com/sap/me/example/view` on the file system relative to the web root.

3.1.4.1 Example View

```
sap.ui.jsview("com.sap.me.example.view.Example", {  
  
    // function called to get name of controller for this view  
    getControllerName: function() {  
        return "com.sap.me.example.view.Example";  
    },  
  
    // function called before view is displayed  
    onBeforeShow : function(evt) {  
        this.getController().onBeforeShow(evt);  
    },  
  
    // function called after view is displayed  
    onAfterShow : function(evt) {  
        this.getController().onAfterShow(evt);  
    },  
  
    // function called to create content  
    // - oCon is reference to controller  
    createContent : function(oCon) {  
  
        // create page  
        this.page = new sap.m.Page({  
            id: this.createId('exampleViewPage'),  
            title : "",  
            icon : sap.ui.core.IconPool.getIconURI("factory"),  
            showNavButton : true,  
            navButtonTap : [ oCon.navButtonTap, oCon ]  
        });  
  
        // list to add controls to  
        var exampleViewList = new sap.m.List({inset : true, mode: sap.m.ListMode.None});
```

```
// create list item to hold controls
var oItem = new sap.m.CustomListItem();
oItem.setType(sap.m.ListType.Inactive);

// Horizontal box to add control to
var oHbox = new sap.m.HBox();
oHbox.setAlignItems(sap.m.FlexAlignItems.Center);

// UI5 control...
var field = new .....({
    id: this.createId('someField'),
    .....
});

// add control to horizontal box container
oHbox.addItem(field);

// add box to list item
oItem.addContent(oHbox);

// add list item to list of control(s)
exampleViewList.addItem(oItem);

// add list of control(s) to page
this.page.addContent(exampleViewList);

return this.page;
}
});
```

3.1.4.2 Example Controller

```
sap.ui.controller("com.sap.me.example.view.Example", {

    // called after controls are created but before displayed
    onBeforeShow : function(evt) {

        // "evt.data.options" will contain any activity options defined to
        // this activity in Activity Maintenance
        if (evt.data && evt.data.options && evt.data.options.length > 0) {
            for (var i=0; i<evt.data.options.length; i++) {
                var option = evt.data.options[i];
                if (option.key === "SOME_KEY") {
                    if (option.value && (option.value === "NO" || option.value === "FALSE")) {
                        // do something
                    }
                }
            }
        }
    },

    // called after controls are initially displayed.
    onAfterShow : function(evt) {

        // create/update model and/or update view controls based on model
    },
});
```

```
// returns to Main application screen.

navButtonTap : function(evt) {
    var bus = sap.ui.getCore().getEventBus();
    bus.publish("nav", "to", {
        id : "Home"
    });
}

});
```

The example files above provide a basic structure for an SAP ME Mobile activity. The next sections will provide more details on how to perform navigation between views and implement controls and bind them to a model.

3.1.5 View Navigation

In most cases an activity will be comprised of more than one view requiring some way to navigate between the views. In most cases the navigation is in response to some user action such as a button click. The sample view code below shows the definition of a button *Next*. When the user selects the button the controller function `nextTap` will be called.

```
var nextButton = new sap.m.Button({
    id: this.createId('nextButton'),
    type: sap.m.ButtonType.Default,
    text: "Next",
    enabled: true,
    tap : [ oCon.nextTap, oCon ]
});
```

The navigation is accomplished using the SAPUI5 Event Bus to publish a navigation event to the application frameworks Event Bus channel which is called "nav". The second argument to the "publish" function is the event identifier. SAP ME Mobile supports three events; "to", "back" and "home". The last argument to "publish" is a parameter map containing some required parameters as well as optional parameters. The parameter map will be passed in the event that is sent to the view being navigated to. The following code will cause the navigation from the "Example" view to the "NextExample" view.

```
nextTap : function(evt) {
    var bus = sap.ui.getCore().getEventBus();
    bus.publish("nav", "to", {
        id : "NextExample",
        data : {
            namespace : "com.sap.me.example.view",
            fromId : "Example",
            fromNamespace : "com.sap.me.example.view",
            customKey : "someValue",
            customData : oCustomData
        }
    });
}
```

SAP ME 15.1 Mobile Developers Guide

The parameter map must contain an “id” which is the name of the view and the parameter map “data” which contains information required by the application navigation handler. Custom data can also be passed to the view being navigated to. The following table describes the possible parameters that could be part of the map.

Key	Description	Comments
namespace	Namespace for the “to” view	Required
fromId	Name of the “from” view	Optional
fromNamespace	Namespace for the “from” view	Required if fromId used.
applicationId	Name of workstation (POD)	Passed from main view to activity view
workstation	Name of workstation	Passed from main view to activity view
activityRef	Activity reference	Passed from main view to activity view
activityType	<i>Activity Type from Activity Maintenance.</i>	Passed from main view to activity view
options	List of activity options from <i>Activity Maintenance.</i>	Passed from main view to activity view
appName	Name of activity from <i>Activity Maintenance</i>	Passed from main view to activity view

Additional information to be passed from one view to another can be added to the “data” map like is shown in the example above where customKey and customData will be included in the event object that is passed to the *NextExample* views controller as shown in the following code example.

```
onBeforeShow : function(evt) {  
    // evt.data will contain parameters passed to navigation handler  
    if (evt.data.customKey) {  
    }  
    if (evt.data.customData) {  
    }  
}
```

To return to the SAP ME Mobile “Home” screen you can code the following:

```
var bus = sap.ui.getCore().getEventBus();  
bus.publish("nav", "to", {  
    id : "Home"  
});
```

3.1.6 Data Binding and OData Services

The SAPUI5 Developers Guide provides an extensive description on Models and Data Binding. This section will cover how SAP ME Mobile uses OData Services to communicate with the server. The OData Services Guide contains detailed descriptions of the service requests supported. The `util.IOUtil` class contains a number of helpful functions for making requests to the server. The following is a brief description of the functions used.

Function	Description
<code>getODataRequestResults</code>	Performs a synchronous request to an OData service and returns a JSON object containing the results of request.
<code>getEntitySetModel</code>	Performs a synchronous request to an OData service using an <code>EntitySet</code> name and returns a JSON object containing an array of objects.
<code>remoteRequest</code>	Performs a synchronous request to the server and returns a JSON object containing the results of request.

3.1.6.1 Retrieving data from Server

The following example shows getting a list of Operations from the server. The result is returned as a `JSONModel`.

```
var sFilterOperator = undefined;
if (!util.StringUtil.isBlank(operation)) { // operation might contain a partial name (i.e.; OP) to filter by.
    sFilterOperator = "operation eq '" + operation + "'";
}
var oModel = util.IOUtil.getEntitySetModel("/manufacturing-odata/Production.svc/", "Operations", sFilterOperator);
if (oModel) {
    sap.ui.getCore().setModel(oModel, "operationsModel");
}
```

This next example shows how to get JSON data using the `getODataRequestResults` function.

```
var url = "/manufacturing-odata/Production.svc/BrowseWorkCenters";
if (!util.StringUtil.isBlank(sFilterValue)) { // optional filter value
    url = url + "?WorkCenterMask='" + sFilterValue + "'";
}
var oResults = undefined;
try {
    oResults = util.IOUtil.getODataRequestResults(url, false);
} catch (err) {
    jQuery.sap.log.error(err.message);
    return;
}
```

```
}  
  
// create and load JSONModel  
var oModel = new sap.ui.model.json.JSONModel();  
var oBinding = {};  
oBinding["WorkCenters"] = oResults;  
oModel.setData(oBinding);  
sap.ui.getCore().setModel(oModel, "workCentersModel");
```

3.1.6.2 Sending transaction request to Server

In many cases you want to make a request to the server to perform some transaction. The following example will start an SFC by calling the OData StartSfc service.

```
startSfcTap : function(evt) {  
    var url = "/manufacturing-odata/Production.svc/StartSfc?Sfc='SFC001'&Operation='OPER1'&Resource='DEFAULT'";  
    util.IOUtil.remoteRequest(url, "POST", null, this.startSuccessCallback, this.errorCallback, this);  
},  
startSuccessCallback : function(oData) {  
    sap.m.MessageToast.show("SFC001 Started", {  
        duration: 3000,  
        animationDuration: 500  
    });  
},  
errorCallback : function(errorCode, errorMessage) {  
    if (errorMessage) {  
        sap.m.MessageBox.alert(errorMessage);  
    }  
}
```

3.1.7 Localized Text

SAP ME Mobile uses an OData service for getting localized text from resource files on the server. The `util.I18NUtility.getLocaleSpecificText` utility class can be used as follows to return text in the correct locale:

```
var sLocalizedText = util.I18NUtility.getLocaleSpecificText("EXAMPLE.default.LABEL");
```

3.1.8 Extended Text Controls

SAP ME Mobile provides two extended text type controls that provide additional features than the standard SAPUI5 mobile controls. The `com.sap.me.control.Input` control can be used in place of `sap.m.Input` and the `com.sap.me.control.TextArea` can be used in place of the `sap.m.TextArea` control.

3.1.8.1 Input Control

The extended input control is based on the `sap.m.Input` control with the added functionality to add a clear icon that can be used to clear the input field and a browse icon to trigger a call to a controller function. You can also specify if the input field contents should be rendered in upper case.



```
var operInputField = new com.sap.me.control.Input({
  id: this.createId('operationInput'),
  maxLength: 40,
  width: "256px",
  placeholder: util.I18NUtility.getLocaleSpecificText("OPERATION_BROWSE.default.LABEL"),
  showClear: true,
  showBrowse: true,
  upperCase: true,
  browseTap: [ oCon.browseOperationTap, oCon ]
});
```

3.1.8.2 TextArea Control

The extended text area control is based on the `sap.m.TextArea` control with the added functionality to add a clear icon that can be used to clear the text field. It also provides the ability to provide a value state text to display next to the control if an error occurs as well as a flag to describe the type of value state message to show.

```
var textAreaField = new com.sap.me.control.TextArea({
  id: oView.createId("exampleText"),
  width: "100%",
  rows: 3,
  showClear: true
});
```

If an error occurs at some point you can display a message next to the component using the following control methods.

```
textAreaField.setValueStateText(errorMessage);
textAreaField.setValueState(sap.ui.core.ValueState.Error);
```

3.2 SAP ME Common Utility Functions

There are a number of utilities that are provided in SAP ME to help with the building of mobile activities. These utilities are available to any SAPUI5 activities, not just SAP ME Mobile.

- `util.IOUtil` – Functions for making server requests.
- `util.I18NUtility` – Functions for getting locale specific text or error messages.
- `util.Message` – Functions for displaying messages on the device.
- `util.Dialog` – Functions for displaying dialogs on the device.
- `util.Activity` – Functions for retrieving activity rules and configuration information.
- `util.SystemRules` – Functions for retrieving System Rules from the server.
- `util.Model` – Functions for retrieving and storing data in local storage.
- `util.StringUtil` – Functions for working with String data.
- `util.CommonKey` – Common constants for storing shared information in the Model.

3.2.1 `util.IOUtil` Functions

```
/**
 * Will perform a synchronous request using the input servletUrl.
 *
 * @param servletUrl url of servlet
 * @param requestType Must be either "POST" or "GET"
 * @param requestData Optional JSON object to post in request
 * @param successCallback optional function called with response JSON data:
 * <pre>
 *   callback(oData)
 * </pre>
 * @param errorCallback Optional function called when error occurs:
 * <pre>
 *   callback(errorCode, errorMessage)
 * </pre>
 * @param thisArg Optional context to call success and error functions with
 * @return
 */
remoteRequest : function(servletUrl, requestType, requestData, successCallback, errorCallback, thisArg);
```

```
/**
 * Will perform a synchronous request to an odata service and return the results.
 *
 * @param sPath path to odata service
 * @param bGetAllData true to get all data including meta data, false to get "results" portion of response only
 * @return JSON object containing result
 * @throws Error if request returns exception
 */
getODataRequestResults : function(sPath, bGetAllData);

/**
 * Will return an array of objects by an EntitySet name in a odata service and return results as a JSONModel
 *
 * @param {string} sService URL of service (i.e.; "/manufacturing-odata/Production.svc/")
 * @param {string} sName Name of EntitySet to get (i.e.; "Operations")
 * @param {string} sFilterOperator Optional string in form of " operation eq 'OPER' "
 * @param {string} sQueryParameters Optional query parameters to append to end of url
 * @return {sap.ui.model.json.JSONModel} object containing result if found, else undefined
 */
getEntitySetModel : function(sService, sName, sFilterOperator, sQueryParameters);

/**
 * loads a given CSS file
 *
 * @param {String} sURL Uniform resource locator of CSS file to load.
 */
loadCSS : function(sURL);

/**
 * loads a given JavaScript file
 *
 * @param {String} sURL Uniform resource locator of JavaScript file to load.
 */
loadJS : function(sURL);

/**
 * Redirects to given URL
 *
 * @param {String} sURL Uniform resource locator.
 * @param {Boolean} [bNewWindow] Opens URL in a new browser window
 * @public
 */
```

```
* @name util.IOUtil#redirect
* @function
*/
redirect : function (sURL, bNewWindow);
```

3.2.2 util.I18NUtility Functions

```
/**
 * Returns a locale-specific string value for the given key sKey.
 *
 * The text is searched in this resource bundle according to the fallback chain described in
 * {@link jQuery.sap.util.ResourceBundle}. If no text could be found, the key itself is used as text.
 *
 * If text parameters are given, then any occurrences of the pattern "<i>n</i>" with <i>n</i> being an integer
 * are replaced by the parameter value with index <i>n</i>. Note: This replacement is also applied if no text had been found (key).
 *
 * @param {string} sKey
 * @param {jQuery.sap.util.Properties} [oProperties] Map parameters which should replace the
 *      place holders "%...%" tokens in the found locale-specific string value.
 * @return {string} The value belonging to the key, if found; otherwise the key itself.
 */
getLocaleSpecificText : function(sKey, oProperties);

/**
 * Returns a locale-specific string value for the given error key sKey.
 *
 * The text is searched in this resource bundle according to the fallback chain described in
 * {@link jQuery.sap.util.ResourceBundle}. If no text could be found, the key itself is used as text.
 *
 * If text parameters are given, then any occurrences of the pattern "%<i>s</i>%" with <i>s</i> being a string value
 * are replaced by the parameter value with key "%<i>s</i>". Note: This replacement is also applied if no text had been found (key).
 *
 * @param {string} sKey key in bundle to get value for
 * @param {jQuery.sap.util.Properties} [oProperties] Map parameters which should
 *      replace the place holders "%...%" tokens in the found locale-specific string value.
 * @return {string} The value belonging to the key, if found; otherwise the key itself.
 */
getErrorText : function(sKey, oProperties);
```

3.2.3 util.Message Functions

```
/**
 * creates and adds a floating message box to the screen
 *
 * @param {string} sMessage Message to show
 * @param {string} sTitleKey Optional key to title text
 */
showMessageBox : function(sMessage, sTitleKey);

/**
 * closes the message box
 */
closeMessageBox : function();

/**
 * creates and adds a floating message toast to the screen
 *
 * @param {string} sMessage Message to show
 * @param {integer} iDelay number of milliseconds to display
 * @param {string} sTitleKey Optional key to title text
 */
showMessageToast : function(sMessage, iDelay, sTitleKey)
```

3.2.4 util.Dialog Functions

```
/**
 * creates and displays a floating dialog
 *
 * @param {string} sContent HTML to render in the dialog
 * @param {string} sDialogBkgColor background color for dialog
 * @param {boolean} bShowCloseIcon true to show close icon, else false
 * @param {function} fnAfterShow Optional function to call after dialog is displayed
 * @param {object} oFnContext Optional context for function
 */
showDialog : function(sContent, sDialogBkgColor, bShowCloseIcon, fnAfterShow, oFnContext);

/**
 * closes the message box
 */
closeDialog : function();
```

3.2.5 util.Activity Functions

```
/**
 * Gets the activity rules for the input activity id
 *
 * @param {string} sActivityId Activity to get rules for
 * @return array of rules "{attribute: "&sRule", value : &sRuleValue}", undefined if sActivityId is not defined
 * @public
 */
getRules : function(sActivityId);

/**
 * Gets the activity configuration for the input activity id
 *
 * @param {string} sActivityId Activity to get rules for
 * @return activity configuration if found, else undefined
 * <pre>
 * {
 *   "activityRef": activity reference,
 *   "activityId": activity ID,
 *   "activityType": activity type,
 *   "description": activity description,
 *   "classOrProgram": activity class or program
 * }
 * </pre>
 * @throws error if activity not found or no permissions to activity
 * @public
 */
getConfiguration : function(sActivityId);

/**
 * Gets the Mobile activity configuration for the input activity id
 *
 * @param {string} sActivityId Activity to get rules for
 * @return activity configuration if found, else undefined
 * <pre>
 * {
 *   "activityRef": activity reference,
 *   "activityId": activity ID,
 *   "activityType": activity type,
```

```
*      "description": activity description,
*      "namespace": activity namespace,
*      "view": activity view name
*    }
* </pre>
* @throws error if activity not found, no permissions to activity or not supported by mobile
* @public
*/
getMobileConfiguration : function(sActivityId);
```

3.2.6 util.SystemRules Functions

```
/**
 * Returns the system rule for the given key sKey.
 *
 * @param {string} sKey
 * @return {string} The system rule belonging to the key, if found; otherwise undefined if not found
 */
getRule : function(sKey);

/**
 * Returns the system rules for the keys passed in. Can optionally pass an array
 * of contexts in case you have overrides at the context specified.
 *
 * @param {array} aKeys Array of keys to get rules for
 * @param {array} aContexts Optional array of contexts in case you have overrides at the context specified
 * @return {array} Array of objects "{attribute: "&sKey", value : &value}", undefined if no key exists
 */
getRules : function(aKeys, aContexts);
```

3.2.7 util.Model Functions

```
/**
 * Will get JSON data from local storage using input key.
 *
 * @param key unique key to use to store data to
 */
getData : function(key);
```

```
/**
 * Will store data into local storage using input key.
 *
 * @param key unique key to use to store data to
 * @param data JSON formatted data to store
 */
setData : function(key, data);

/**
 * Will remove data into local storage using input key.
 *
 * @param key unique key to use to remove data
 */
removeData : function(key);

/**
 * Will remove all local storage data.
 */
clearAllData : function();

/**
 * Sets whether unsaved data currently exists in model
 *
 * @param bUnsaved boolean true if unsaved data exists, else false
 */
setUnsavedDataDefined : function(bUnsaved);

/**
 * Sets whether unsaved data currently exists in model
 *
 * @return true if unsaved data exists; else false
 */
isUnsavedDataDefined : function();

/**
 * Gets the data assigned to a
 *
 * @param oData array of {sap.ui.core.CustomData} objects
 * @param sKey Key of data to get
 * @return if found returns the value associated with key, else undefined if not found
 */
getCustomDataValue : function(oData, sKey);
```

```
/**
 * Sets the data assigned to a
 *
 * @param oData array of {sap.ui.core.CustomData} objects
 * @param sKey Key of data to set
 * @param oValue data to set into custom data
 */
setCustomDataValue : function(oData, sKey, oValue);

/**
 * Finds the {sap.ui.core.CustomData} object assigned to the key
 *
 * @param oData array of {sap.ui.core.CustomData} objects
 * @param sKey Key of data to get
 * @return {sap.ui.core.CustomData} if found, else undefined if not found
 */
findCustomData : function(oData, sKey);
```

3.2.8 util.StringUtil Functions

```
/**
 * helper function to see if input string is blank
 *
 * @param str {String} to check
 * @return true if string is blank (undefined, null or only spaces), else true
 */
isBlank : function(str);

/**
 * helper function to trim leading and trailing spaces from a string
 *
 * @param str {String} to trim spaces from
 * @return updated String
 */
trimString : function(str);

/**
 * helper function to encode the input string (including single quotes)
 *
 * @param {String} str String to encode
```

```
* @return encoded string
*/
encodeURIComponent : function(str);

/**
 * helper function to check if one string ends with another
 *
 * @param {String} str String to check
 * @param {String} searchString String to search
 * @return true if str ends with sSearch
 */
endsWith : function(str, sSearch);

/**
 * This utility method substitutes in the String _orig all occurrences of string
 * _to substr with the string _withsubst
 *
 * @param _orig The string to search
 * @param _to substr what to search for
 * @param _withsubst what to substitute with
 * @param _startIndex start replacing from this index
 * @return String new string with substitution executed
 */
replaceStrings : function(_orig, _to substr, _withsubst, _startIndex);
```

3.2.9 util.CommonKey Constants

```
/**
 * ModelKey that holds the current site the user is operating on.
 */
```

3.3 SAP ME Mobile Utility Functions

There are also utilities specific to SAP ME Mobile applications to help with the building of mobile activities.

- `util.Common` – Functions for retrieving and saving common “input” type and operation values.
- `util.Navigation` – Functions for transferring control to the input activity.
- `util.ModelKey` – Constants for retrieving and storing shared SAP ME Mobile information in the Model.

3.3.1 util.Common Functions

```
/**
 * Save the InputType and InputTypeValue
 *
 * @param {object} commonInputType - The common Input Type
 * @param {object} commonInputTypeValue The common Input Type Value
 * @public
 */
saveCommonSettings: function (objCommonInputType, objCommonInputTypeValue, objCommonOperation, objCommonResource);

/**
 * Apply the InputType and InputTypeValue to their respective object (from the previously saved model)
 *
 * @param {object} commonInputType - The common Input Type
 * @param {object} commonInputTypeValue The common Input Type Value
 * @public
 */
applyCommonSettings: function (objCommonInputType, objCommonInputTypeValue, objCommonOperation, objCommonResource);
```

3.3.2 util.Navigation Functions

```
/**
 * transfer control to input activity
 *
 * @param {string} sActivityId Activity to go to
 * @param {string} sFromId View ID of caller
 * @param {string} sNamespace Namespace of caller
 * @param {object} oData Optional data object to pass
 * @throw Error if failure to go to activity
 * @public
 */
gotoActivity : function(sActivityId, sViewId, sNamespace, oData);
```

3.3.3 util.ModelKey Constants

```
/**
 * ModelKey Workstation holds name of workstation loaded
 */
/**
 * ModelKey WorkstationConfiguration holds JSON object holding workstation configuration information
 */
```

```
/**
 * ModelKey CurrentApplicationView holds name of current active application view (Not sub-views)
 */
/**
 * ModelKey CurrentApplicationNamespace holds name of current active application namespace (Not sub-views)
 */
/**
 * ModelKey CurrentApplicationRules holds current application "activity rules"
 */
/**
 * ModelKey SelectedOperation holds currently selected Operation
 */
/**
 * ModelKey SelectedOperationRevision holds currently selected Operation revision
 */
/**
 * ModelKey SelectedResource holds currently selected Resource
 */
/**
 * ModelKey SelectedComponent holds currently selected Component
 */
/**
 * ModelKey SelectedAssemblyOperation holds the Operation assignment for the current Component
 */
/**
 * ModelKey SelectedComponentRevision holds currently selected Component revision
 */
/**
 * ModelKey SelectedComponentDescription holds currently selected Component description
 */
/**
 * ModelKey SelectedLotSize holds currently selected Lot Size
 */
/**
 * ModelKey SelectedComponentQuantity holds currently selected Component quantity
 */
/**
 * ModelKey SelectedHasAssyData holds a boolean indicating if the selected component contains assembly data fields
 */
/**
 * ModelKey SelectedAssemblyData holds a JSON string of Selected assembly Data
```

```
*/
/**
 * ModelKey SelectedDefaultComponentQuantity holds the Quantity value that will appear in the text box.
 */
/**
 * ModelKey SelectedSFC holds currently selected SFC
 */
/**
 * ModelKey SelectedTypeCount holds current count of SFC's.
 */
/**
 * ModelKey SelectedNCGroup holds currently selected NC Group
 */
/**
 * ModelKey SelectedNCCode holds currently selected NC Code
 */
/**
 * ModelKey SelectedCollectionType holds the type data is being collected for (SFC, SHOP_ORDER or PROCESS_LOT)
 */
/**
 * ModelKey SelectedCollectionTypeValue holds the value for the type of data being collected
 */
/**
 * ModelKey SelectedDCGroupList holds currently selected DC Group(s)
 */
/**
 * ModelKey SelectedDCGroupList holds currently selected DC Group(s)
 */
/**
 * ModelKey SelectedEquipmentChangeType holds currently selected Type of Equipment Change
 */
/**
 * ModelKey SelectedToolGroup holds currently selected Tool Group
 */
/**
 * ModelKey SelectedToolNumber holds currently selected Tool Number
 */
/**
 * ModelKey SelectedWorkCenter holds currently selected Tool Work Center
 */
/**
```

```
* ModelKey SelectedStatusCode holds currently selected Status Code
*/
/**
* ModelKey SelectedStatusCode holds currently selected Status Code
*/
/**
* ModelKey SelectedReasonCode holds currently selected Reason Code
*/
/**
* ModelKey SelectedReasonCodeCategory holds currently selected Category of Reason Code
*/
```

4 SAP ME Mobile “Lean”

SAP ME Mobile “Lean” is an example application for use on older devices that do not have browsers that support HTML5 as well as Window CE devices. These devices are not able to run applications that use SAPUI5 controls. They still are able to use many utilities that are included with SAPUI5 by using the SAPUI5 “lean” library (i.e.; sap-ui-core-lean.js). The example application provided is defined to allow browsing for an operation and resource as well as the entering of an SFC value. It then will provide start, signoff and complete transactions on the server.

The following sample html provides the minimum requirement for a page that is able to access OData services and has access to the lean libraries as provided by SAPUI5. It does not support the use of any SAPUI5 UI controls nor should it include html5 markup.

```
<%
int iDeviceWidth = -1;
String sDeviceWidth = request.getParameter("DEVICE_WIDTH");
if (sDeviceWidth != null && sDeviceWidth.trim().length() > 0) {
    try {
        iDeviceWidth = Integer.parseInt(sDeviceWidth);
    } catch (Exception e) {
    }
}
}%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<%
    if (iDeviceWidth <= 0) {
%>
        <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=2.0, user-scalable=yes" />
<%
    } else {
%>
        <meta name="viewport" content="width=<%=iDeviceWidth%>, initial-scale=1.0, maximum-scale=2.0, user-scalable=yes" />
        <meta name="MobileOptimized" content="<%=iDeviceWidth%>">
<%
    }
%>
<title>Example Title</title>
```

```
<script id="sap-ui-bootstrap" type="text/javascript"
  src="/sapui5/resources/sap-ui-core-lean.js">
</script>

<link rel="stylesheet" type="text/css" href="../css/message.css">
<link rel="stylesheet" type="text/css" href="css/base/style.css">
<link rel="stylesheet" type="text/css" href="css/base/browse.css">
<link rel="stylesheet" type="text/css" href="css/base/message.css">

<style type="text/css">
  html, body {
    height: 100%;
    background-color: rgb(222, 222, 222);
  }
  html{
    height: 100%;
  }
  body {
    margin: 0px;
    min-height: 100%;
  }
  .pageContainer {
    display: none;
    height: 100%;
    overflow: hidden;
    overflow-x: hidden;
    width: 100%;
  }
  .noScriptMessage {
    top: 30%;
    left: 10%;
    right: 10%;
    height: auto;
    line-height: 44px;
    position: absolute;
    font-size: 24px;
    font-weight: bold;
    font-family: 'Helvetica Neue', Roboto, 'Droid Sans', 'BBGlobal Sans', 'Segoe WP', Helvetica, Arial, sans-serif;
    color: red;
  }
</style>

<script type="text/javascript">

  // Load common ui5 utilities
  sap.ui.localResources("com.sap.me.ui5.util");
  jQuery.sap.registerModulePath('util', '../com/sap/me/ui5/util');

  sap.ui.localResources("../lean");
  jQuery.sap.registerModulePath("lean", "../lean");
  jQuery.sap.require("util.StringUtil");

  var sSize = '<%=request.getParameter("SIZE")%>';
  if (util.StringUtil.isBlank(sSize) || sSize === 'null') {
    sSize = '<%=request.getParameter("size")%>';
  }
  if (util.StringUtil.isBlank(sSize) || sSize === 'null') {
    sSize = 'small';
  }

  // load query parameter information
  this.sSizeUI = "LARGE";
  this.sCssPath = "css/large/";
  if (sSize.toLowerCase() === "medium") {
    this.sSizeUI = "MEDIUM";
```

```
this.sCssPath = "css/medium/";
} else if (sSize.toLowerCase() === "small") {
    this.sSizeUI = "SMALL";
    this.sCssPath = "css/small/";
}

this.sSite = '<%=request.getParameter("SITE")%>';
if (util.StringUtil.isBlank(this.sSite) || this.sSite === 'null') {
    this.sSite = '<%=request.getParameter("site")%>';
}
if (this.sSite === 'null') {
    this.sSite = undefined;
}

this.sDefaultOperation = '<%=request.getParameter("OPERATION")%>';
if (util.StringUtil.isBlank(this.sDefaultOperation) || this.sDefaultOperation === 'null') {
    this.sDefaultOperation = '<%=request.getParameter("operation")%>';
}
if (this.sDefaultOperation === 'null') {
    this.sDefaultOperation = undefined;
}

this.sDefaultResource = '<%=request.getParameter("RESOURCE")%>';
if (util.StringUtil.isBlank(this.sDefaultResource) || this.sDefaultResource === 'null') {
    this.sDefaultResource = '<%=request.getParameter("resource")%>';
}
if (this.sDefaultResource === 'null') {
    this.sDefaultResource = undefined;
}

// set to true to enable animation for dialogs and messages
this.bUseAnimation = true;
var sNoAnimation = '<%=request.getParameter("NO_ANIMATION")%>';
if (util.StringUtil.isBlank(sNoAnimation) || sNoAnimation === 'null') {
    sNoAnimation = '<%=request.getParameter("no_animation")%>';
}
if (!util.StringUtil.isBlank(sNoAnimation) && sNoAnimation === 'true') {
    this.bUseAnimation = false;
}

jQuery.sap.require("util.IOUtil");
jQuery.sap.require("util.Model");
jQuery.sap.require("util.CommonKey");
jQuery.sap.require("util.I18NUtility");
jQuery.sap.require("util.Message");
jQuery.sap.require("util.Dialog");
jQuery.sap.require("lean.Browse");

// Called when the document is displayed. It forwards to the afterShow() function
function initializeForm() {
    var thisController = this;
    $( document ).ready(function() {
        thisController.afterShow();
    });
}

function afterShow() {

    // If scripting is not allowed, this will cause the No Script message to display
    $("#noscriptmsg").css("display", "none");
    $("#pagecontainer").css("display", "block");

    // loads the appropriate CSS for the size specified in query parameter
    util.IOUtil.loadCSS(this.sCssPath + "style.css");
    util.IOUtil.loadCSS(this.sCssPath + "browse.css");
}
```

```
util.IOUtil.loadCSS(this.sCssPath + "message.css");

// load message css for non-animated popups
if (this.bUseAnimation === false) {
    util.IOUtil.loadCSS("../lean/css/base/message_passive.css");
}

$("#headerIcon").attr("src", this.sCssPath + "img/info.png");
$("#operationBrowseIcon").attr("src", this.sCssPath + "img/action.png");
$("#resourceBrowseIcon").attr("src", this.sCssPath + "img/action.png");

// Must change site first so we get correct operation / resource data
try {
    if (!util.StringUtil.isBlank(sSite)) {
        util.Model.setData(util.CommonKey.CurrentSite, sSite);
    } else {

        // Find the site for the current user. This will be the default site. If there is no default, the first
        // site the user is defined within is obtained.
        var curSite = util.IOUtil.getODataRequestResults("/manufacturing-odata/AppConfig.svc/GetSiteForUser", true);
        if (curSite) {
            util.Model.setData(util.CommonKey.CurrentSite, curSite.GetSiteForUser.site);
        } else {
            jQuery.sap.log.error("Unable to find a site for the current user.");
            return;
        }
    }
} catch (err) {
    jQuery.sap.log.error("afterShow: " + err.message);
}

// sets up event handler to display information about the browser
$(document).on("click", "#headerIcon", function(e) {
    var windowWidth = window.screen.width < window.outerWidth ? window.screen.width : window.outerWidth;
    var windowHeight = window.screen.height < window.outerHeight ? window.screen.height : window.outerHeight;
    if (!windowWidth || !windowHeight) {
        if (document.body && document.body.offsetWidth) {
            windowWidth = document.body.offsetWidth;
            windowHeight = document.body.offsetHeight;
        } else if (document.compatMode==='CSS1Compat' && document.documentElement &&
            document.documentElement.offsetWidth) {
            windowWidth = document.documentElement.offsetWidth;
            windowHeight = document.documentElement.offsetHeight;
        }
    }
    var message = "Width=" + windowWidth + ", Height =" + windowHeight + "<br>";
    message = message + "User Agent=" + navigator.userAgent + "<br>";
    util.Message.showMessageBox(message, "Information");
});

// WIP Template Section 2: Implement Controls

}

// This is called as soon as the page is loaded
(function() {
    initializeForm();
})();

</script>

</head>
```

```
<body id="content">

  <div id="pagecontainer" class="pageContainer">

    <div id="headercontainer" class="headerRow">
      <label id="headerLabel" class="headerText">WIP Processing</label>
      <img id="headerIcon" class="headerInfoIcon"></img>
    </div>

    <div id="inputcontainer" class="inputControlRows" >

      // WIP Template Section 1: Control Creation - Layout controls here

    </div>

    <div id="toolbar" class="toolbarRow">
      // WIP Template Section 3: Toolbar Creation - Add toolbar buttons here
    </div>
  </div>

  <div id="noscriptmsg" class="noScriptMessage">
    JavaScript is not enabled and is required to run WIP.
  </div>
</body>
</html>
```

4.1 Application Example

The example above provides a template that can be used to start a new “lean” application from. In the template above there are 3 sections that need to be completed to build a working application. Once you have designed what the user interface will look like you can start adding the controls.

4.1.1 Control Creation

The controls must be added to the page. In the template above, find the section called “WIP Template 1” and insert the following code. This adds 3 rows of input fields, two of which are “browse” fields.

```
<div class="inputControlFields" >
  <div class="leftColumn">
    <div class="labelContainer">
      <label id="operationLabel" for="operationField" class="truncatedLabel fieldLabel"></label>
    </div>
  </div>
  <div class="rightColumn">
    <div class="browseContainer">
      <div class="browseFieldContainer">
        <input id="operationField" class="inputUpperCase operationField" type="text" ></input>
      </div>
      <div class="browseButtonContainer">
        <button id="operationBrowseButton" class="browseButton" type="button" >
          <img id="operationBrowseIcon" class="browseButtonIcon"></img>
        </button>
      </div>
    </div>
  </div>
  <div class="leftColumn">
    <div class="labelContainer">
      <label id="resourceLabel" for="resourceField" class="truncatedLabel fieldLabel"></label>
    </div>
  </div>
```

```
</div>
<div class="rightColumn">
  <div class="browseContainer">
    <div class="browseFieldContainer">
      <input id="resourceField" class="inputUpperCase resourceField" type="text"></input>
    </div>
    <div class="browseButtonContainer">
      <button id="resourceBrowseButton" class="browseButton" type="button">
        <img id="resourceBrowseIcon" class="browseButtonIcon"></img>
      </button>
    </div>
  </div>
</div>
<div class="leftColumn">
  <div class="labelContainer">
    <label id="sfcLabel" for="sfcField" class="truncatedLabel fieldLabel"></label>
  </div>
</div>
<div class="rightColumn">
  <input id="sfcField" class="inputUpperCase sfcField" type="text" ></input>
</div>
</div>
```

4.1.2 Toolbar Creation

The toolbar controls are then added to the page. In the template above, find the section called “WIP Template 2” and insert the following code. This adds 3 buttons to the Toolbar.

```
<button id="startButton" class="toolbarButton" type="button"></button>
<button id="signoffButton" class="toolbarButton" type="button"></button>
<button id="completeButton" class="toolbarButton" type="button"></button>
```

4.1.3 Implement Controls

The next step is to implement the control behavior. In the template above, find the section called “WIP Template 3” and insert the following code. Comments in the code will provide more detailed information.

```
var thisController = this;
try {

  // set up event handler to display information about the browser
  $(document).on("click", "#headerIcon", function(e) {
    thisController.showBrowserInformation();
  });

  // get the Locale labels
  var sStartLabel = util.I18NUtility.getLocaleSpecificText("WIP_PROCESSING.startButton.LABEL");
  var sSignoffLabel = util.I18NUtility.getLocaleSpecificText("WIP_PROCESSING.signoffButton.LABEL");
  var sCompleteLabel = util.I18NUtility.getLocaleSpecificText("WIP_PROCESSING.completeButton.LABEL");
  var sOperationLabel = util.I18NUtility.getLocaleSpecificText("OPERATION_BROWSE.default.LABEL");
  var sResourceLabel = util.I18NUtility.getLocaleSpecificText("RESOURCE_BROWSE.default.LABEL");
  var sSfcLabel = util.I18NUtility.getLocaleSpecificText("SFC_BROWSE.default.LABEL");

  // Add text and “onclick” behavior to the toolbar buttons. The text is obtained
  // using OData services for the correct Locale.
  var oButton = $("#startButton");
  if (oButton) {
    oButton.text(sStartLabel);
    $(document).on("click", "#startButton", function(e) { thisController.start(); });
  }
  oButton = $("#signoffButton");
```

```
if (oButton) {
    oButton.text(sSignoffLabel);
    $(document).on("click", "#signoffButton", function(e) { thisController.signoff(); });
}
oButton = $("#completeButton");
if (oButton) {
    oButton.text(sCompleteLabel);
    $(document).on("click", "#completeButton", function(e) { thisController.complete(); });
}

// Add Locale specific text to the labels used for the Operation, Resource and SFC fields.
$("#operationLabel").text(sOperationLabel);
$("#resourceLabel").text(sResourceLabel);
$("#sfcLabel").text(sSfcLabel);

// Add "onclick" behaviour for the Operation and Resource "browse" buttons.
$(document).on("click", "#operationBrowseButton", function(e) { thisController.browseOperation(); });
$(document).on("click", "#resourceBrowseButton", function(e) { thisController.browseResource(); });

} catch (err) {
    jQuery.sap.log.error("afterShow: " + err.message);
    util.Message.showMessageBox(err.message, "error.title.TEXT");
}

// The Operation and Resource fields can be assigned default values as query parameters
// in the URL. This section of code obtains the default values and then calls an OData
// service to validate they are valid values.
try {
    var sDefaultOperation = jQuery.sap.getUriParameters().get("OPERATION");
    if (sDefaultOperation == undefined || sDefaultOperation == null) {
        sDefaultOperation = jQuery.sap.getUriParameters().get("operation");
    }
    if (sDefaultOperation) {
        var parameters="?operation=" + sDefaultOperation.toUpperCase() + "&";
        var url = "/manufacturing-odata/Production.svc/ValidateOperation" + parameters;
        util.IOUtil.getODataRequestResults(url, true);
        $("#operationField").val(sDefaultOperation);
    }

    var sDefaultResource = jQuery.sap.getUriParameters().get("RESOURCE");
    if (sDefaultResource == undefined || sDefaultResource == null) {
        sDefaultResource = jQuery.sap.getUriParameters().get("resource");
    }
    if (sDefaultResource) {
        var parameters="?resource=" + sDefaultResource.toUpperCase() + "&";
        var url = "/manufacturing-odata/Production.svc/ValidateResource" + parameters;
        util.IOUtil.getODataRequestResults(url, true);
        $("#resourceField").val(sDefaultResource);
    }

} catch (err) {
    jQuery.sap.log.error(err.message);
    util.Message.showMessageBox(err.message, "error.title.TEXT");
}

// The control setup is finished. The next step must be done after
// the page is displayed so it is triggered after a ¼ second delay.
setTimeout(this.adjustControls, 250);
};

// Function will adjust the toolbar to force it under the input fields
```

// This must be done after page is initially displayed so that the
// offset dimensions are set.

```
function adjustControls() {  
    var oDomInputContainer = jQuery.sap.domById("inputcontainer");  
    if (oDomInputContainer) {  
        var offsetTop = oDomInputContainer.offsetTop;  
        var offsetHeight = oDomInputContainer.offsetHeight;  
        var oToolbar = $("#toolbar");  
        if (oToolbar) {  
            oToolbar.css("top", offsetTop + offsetHeight + 10); // 10 for padding  
        }  
    }  
};
```

// "onclick" handler for info icon

```
function showBrowserInformation() {  
  
    var windowWidth = window.screen.width < window.outerWidth ? window.screen.width : window.outerWidth;  
    var windowHeight = window.screen.height < window.outerHeight ? window.screen.height : window.outerHeight;  
    if (!windowWidth || !windowHeight) {  
        if (document.body && document.body.offsetWidth) {  
            windowWidth = document.body.offsetWidth;  
            windowHeight = document.body.offsetHeight;  
        } else if (document.compatMode==='CSS1Compat' && document.documentElement &&  
document.documentElement.offsetWidth) {  
            windowWidth = document.documentElement.offsetWidth;  
            windowHeight = document.documentElement.offsetHeight;  
        }  
    }  
  
    var message = "Layout Width=" + windowWidth + ", Height=" + windowHeight + "<br>";  
    message = message + "Viewport Width=" + window.innerWidth + ", Height=" + window.innerHeight + "<br>";  
    message = message + "User Agent=" + navigator.userAgent + "<br>";  
    util.Message.showMessageBox(message, "Information", this.bUseAnimation);  
};
```

// "onclick" handler for start button – Calls OData service to start the SFC.

```
function start() {  
    var requestData = this.getRequestData();  
    if (!requestData) {  
        return;  
    }  
    var parameters = "?Sfc=" + requestData.sfc + "&Operation=" + requestData.operation + "&Resource=" +  
requestData.resource + "";  
    var url = "/manufacturing-odata/Production.svc/StartSfc";  
    util.IOUtil.remoteRequest(url + parameters, "GET", null, this.startSuccessCallback, this.errorCallback, this);  
};
```

// "onclick" handler for signoff button – Calls OData service to signoff the SFC.

```
function signoff() {  
  
    var requestData = this.getRequestData();  
    if (!requestData) {  
        return;  
    }  
    var parameters = "?Sfc=" + requestData.sfc + "&Operation=" + requestData.operation + "&Resource=" +  
requestData.resource + "";  
    var url = "/manufacturing-odata/Production.svc/SignoffSfc";  
    util.IOUtil.remoteRequest(url + parameters, "GET", null, this.signoffSuccessCallback, this.errorCallback, this);  
};
```

// "onclick" handler for complete button – Calls OData service to complete the SFC.

```
function complete() {
    var requestData = this.getRequestData();
    if (!requestData) {
        return;
    }
    var parameters = "?Sfc=" + requestData.sfc + "&Operation=" + requestData.operation + "&Resource=" +
requestData.resource + "";
    var url = "/manufacturing-odata/Production.svc/CompleteSfc";
    util.IOUtil.remoteRequest(url + parameters, "GET", null, this.completeSuccessCallback, this.errorCallback, this);
};
```

// helper function to get required values from operation, resource and SFC fields.

```
function getRequestData() {

    var operation = $("#operationField").val();
    if (!operation) {
        var message = util.I18NUtility.getErrorText("12903.simple");
        if (this.bUseAnimation) {
            util.Message.showMessageToast(message, 3000, "error.title.TEXT");
        } else {
            util.Message.showMessageBox(message, "error.title.TEXT", this.bUseAnimation);
        }
        return undefined;
    }
    var resource = $("#resourceField").val();
    if (!resource) {
        var message = util.I18NUtility.getErrorText("12904.simple");
        if (this.bUseAnimation) {
            util.Message.showMessageToast(message, 3000, "error.title.TEXT");
        } else {
            util.Message.showMessageBox(message, "error.title.TEXT", this.bUseAnimation);
        }
        return undefined;
    }
    var sfc = $("#sfcField").val();
    if (!sfc) {
        var message = util.I18NUtility.getErrorText("12906.simple");
        if (this.bUseAnimation) {
            util.Message.showMessageToast(message, 3000, "error.title.TEXT");
        } else {
            util.Message.showMessageBox(message, "error.title.TEXT", this.bUseAnimation);
        }
        return undefined;
    }

    var requestData = {
        operation : operation.toUpperCase(),
        resource : resource.toUpperCase(),
        sfc : sfc.toUpperCase()
    };

    return requestData;
};
```

// callback function if start transaction finishes successfully.

```
function startSuccessCallback(oData) {
    var sMessage = util.I18NUtility.getLocaleSpecificText("ME_MOBILE.sfc_started.TEXT");
    if (this.bUseAnimation) {
        util.Message.showMessageToast(sMessage, 3000);
    } else {
        util.Message.showMessageBox(sMessage, "message.default.LABEL", this.bUseAnimation);
    }
};
```

// callback function if signoff transaction finishes successfully.

```
function signoffSuccessCallback(oData) {
    var sMessage = util.i18NUtility.getLocaleSpecificText("ME_MOBILE.sfc_signoff.TEXT");
    if (this.bUseAnimation) {
        util.Message.showMessageToast(sMessage, 3000);
    } else {
        util.Message.showMessageBox(sMessage, "message.default.LABEL", this.bUseAnimation);
    }
};
```

// callback function if complete transaction finishes successfully.

```
function completeSuccessCallback(oData) {
    var sMessage = util.i18NUtility.getLocaleSpecificText("ME_MOBILE.sfc_completed.TEXT");
    if (this.bUseAnimation) {
        util.Message.showMessageToast(sMessage, 3000);
    } else {
        util.Message.showMessageBox(sMessage, "message.default.LABEL", this.bUseAnimation);
    }
};
```

// callback function if error occurs in start, signoff or complete transactions.

```
function errorCallback(errorCode, errorMessage) {
    if (errorMessage) {
        jQuery.sap.log.debug("WipProcessing Error: " + errorMessage);
        util.Message.showMessageBox(errorMessage, "error.title.TEXT");
    }
};
```

// function will call OData service to get operations and then opens a
// browse dialog to allow selection of a operation.

```
function browseOperation() {

    var url = "/manufacturing-odata/Production.svc/Operations?$skip=0&$top=100&$inlinecount=allpages&$format=json";
    var sOperation = $("#operationField").val();
    if (!util.StringUtil.isBlank(sOperation)) {
        url = url + "&$filter=operation%20eq%20'" + sOperation + "'";
    }
    var oData = undefined;
    try {
        oData = util.IOUtil.getODataRequestResults(url, false);
    } catch (err) {
        jQuery.sap.log.error(err.message);
        if (this.bUseAnimation) {
            util.Message.showMessageToast(err.message, 3000);
        } else {
            util.Message.showMessageBox(err.message, "error.title.TEXT", this.bUseAnimation);
        }
        return;
    }
    if (!oData || oData.length === 0) {
        // 12213.simple=No records found
        var sMessage = util.i18NUtility.getErrorText("12213.simple");
        jQuery.sap.log.info("browseOperation Error: " + sMessage + " for operation '" + sOperation + "'");
        if (this.bUseAnimation) {
            util.Message.showMessageToast(sMessage, 3000);
        } else {
            util.Message.showMessageBox(sMessage, "error.title.TEXT", this.bUseAnimation);
        }
        return;
    }

    var oProperties = [];
    oProperties[0] = "operation";
    oProperties[1] = "i18NDescription";
    lean.Browse.browse(oData, oProperties, "OPERATION_BROWSE.title.TEXT", this.sSizeUI, this.operationBrowseCallback, this,
    this.bUseAnimation);
}
```

```
};

// callback function invoked when user selects a operation in the browse dialog
function operationBrowseCallback(oData) {
    if (oData) {
        $("#operationField").val(oData.operation);
        lean.Browse.close();
    }
};

// function will call OData service to get resources and then opens a
// browse dialog to allow selection of a resource.
function browseResource() {

    var url = "/manufacturing-odata/Production.svc/Resources?$skip=0&$top=100&$inlinecount=allpages&$format=json";
    var sResource = $("#resourceField").val();
    if (!util.StringUtil.isBlank(sResource)) {
        url = url + "&$filter=resource%20eq%20'" + sResource + "'";
    }
    var oData = undefined;
    try {
        oData = util.IOUtil.getODataRequestResults(url, false);
    } catch (err) {
        jQuery.sap.log.error(err.message);
        if (this.bUseAnimation) {
            util.Message.showMessageToast(err.message, 3000);
        } else {
            util.Message.showMessageBox(err.message, "error.title.TEXT", this.bUseAnimation);
        }
        return;
    }
    if (!oData || oData.length === 0) {
        // 12213.simple=No records found
        var sMessage = util.I18NUtility.getErrorText("12213.simple");
        jQuery.sap.log.info("browseOperation Error: " + sMessage + " for resource '" + sResource + "'");
        if (this.bUseAnimation) {
            util.Message.showMessageToast(sMessage, 3000);
        } else {
            util.Message.showMessageBox(sMessage, "error.title.TEXT", this.bUseAnimation);
        }
        return;
    }

    var oProperties = [];
    oProperties[0] = "resource";
    oProperties[1] = "i18NDescription";
    lean.Browse.browse(oData, oProperties, "RESOURCE_BROWSE.title.TEXT", this.sSizeUI, this.resourceBrowseCallback, this,
    this.bUseAnimation);
};

// callback function invoked when user selects a resource in the browse dialog
function resourceBrowseCallback(oData) {
    if (oData) {
        $("#resourceField").val(oData.resource);
        lean.Browse.close();
    }
};
```

5 SAP ME Mobile “Ultra-Lean”

SAP ME Mobile “Ultra-Lean” is an example application for use on older devices that do not have browsers that support JQuery or the SAPUI5 lean libraries. This application is

dependent of the [“datajs.js”](#) library to provide access to OData services. The example application provided is defined to provide the same functionality as that in the ME Mobile “Lean” example described in section 4.

The “datajs.js” library only supports asynchronous calls to the server; thus for all server requests you must provide a callback function that will return results. A set of utilities are defined (“ultralean.js”) that provides much of the same functionality as those in the “Lean” application but modified to support only asynchronous calls. In the example below all utility functions defined in ultralean.js will be highlighted in yellow.

```
<%
  int iDeviceWidth = -1;
  String sDeviceWidth = request.getParameter("DEVICE_WIDTH");
  if (sDeviceWidth != null && sDeviceWidth.trim().length() > 0) {
    try {
      iDeviceWidth = Integer.parseInt(sDeviceWidth);
    } catch (Exception e) {
    }
  }
%>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <%
      if (iDeviceWidth <= 0) {
    %>
      <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=2.0, user-scalable=yes" />
    <%
      } else {
    %>
      <meta name="viewport" content="width=<%=iDeviceWidth%>, initial-scale=1.0, maximum-scale=2.0, user-scalable=yes" />
      <meta name="MobileOptimized" content="<%=iDeviceWidth%>">
    <%
      }
    %>
    <title>WIP Processing</title>

    <script id="datajs" type="text/javascript"
      src="/sapui5/resources/sap/ui/thirdparty/datajs.js">
    </script>

    <script id="utiljs" type="text/javascript"
      src="ultralean.js">
    </script>

    <link rel="stylesheet" type="text/css" href="../../lean/css/base/style.css">
    <link rel="stylesheet" type="text/css" href="../../lean/css/base/browse.css">
    <link rel="stylesheet" type="text/css" href="../../lean/css/base/message.css">
    <link rel="stylesheet" type="text/css" href="../../css/message.css">

    <style type="text/css">

      html, body {
        height: 100%;
        background-color: rgb(222, 222, 222);
      }
      html{
        height: 100%;
      }
      body {
        margin: 0px;
        min-height: 100%;
```

```
}
.pageContainer {
  display: none;
  height: 100%;
  overflow: hidden;
  overflow-x: hidden;
  width: 100%;
}
.noScriptMessage {
  top: 30%;
  left: 10%;
  right: 10%;
  height: auto;
  line-height: 44px;
  position: absolute;
  font-size: 24px;
  font-weight: bold;
  font-family: 'Helvetica Neue', Roboto, 'Droid Sans', 'BBGlobal Sans', 'Segoe WP', Helvetica, Arial, sans-serif;
  color: red;
}
</style>
```

```
<script type="text/javascript">
```

```
// load Query paramters and define global variables
```

```
var sSize = '<%=request.getParameter("SIZE")%>';
if (isBlank(sSize) || sSize === 'null') {
  sSize = '<%=request.getParameter("size")%>';
}
if (isBlank(sSize) || sSize === 'null') {
  sSize = 'small';
}

this.sSizeUI = "LARGE";
this.sCssPath = "css/large/";
if (sSize.toLowerCase() === "medium") {
  this.sSizeUI = "MEDIUM";
  this.sCssPath = "css/medium/";
} else if (sSize.toLowerCase() === "small") {
  this.sSizeUI = "SMALL";
  this.sCssPath = "css/small/";
}

this.sSite = '<%=request.getParameter("SITE")%>';
if (isBlank(this.sSite) || this.sSite === 'null') {
  this.sSite = '<%=request.getParameter("site")%>';
}
if (this.sSite === 'null') {
  this.sSite = undefined;
}

this.sDefaultOperation = '<%=request.getParameter("OPERATION")%>';
if (isBlank(this.sDefaultOperation) || this.sDefaultOperation === 'null') {
  this.sDefaultOperation = '<%=request.getParameter("operation")%>';
}
if (this.sDefaultOperation === 'null') {
  this.sDefaultOperation = undefined;
}

this.sDefaultResource = '<%=request.getParameter("RESOURCE")%>';
if (isBlank(this.sDefaultResource) || this.sDefaultResource === 'null') {
  this.sDefaultResource = '<%=request.getParameter("resource")%>';
}
if (this.sDefaultResource === 'null') {
  this.sDefaultResource = undefined;
}
```

```
}

this.sLocalizedText = '<%=request.getParameter("LOCALIZED_TEXT")%>';
if (isBlank(this.sLocalizedText) || this.sLocalizedText === 'null') {
    this.sLocalizedText = '<%=request.getParameter("localized_text")%>';
}
if (this.sLocalizedText === 'null') {
    this.sLocalizedText = undefined;
}
this.localeTextKeys = [
    "WIP_PROCESSING.startButton.LABEL",
    "WIP_PROCESSING.signoffButton.LABEL",
    "WIP_PROCESSING.completeButton.LABEL",
    "OPERATION_BROWSE.default.LABEL",
    "RESOURCE_BROWSE.default.LABEL",
    "SFC_BROWSE.default.LABEL",
    "error.title.TEXT", null,
    "message.default.LABEL", null,
    "ME_MOBILE.sfc_started.TEXT",
    "ME_MOBILE.sfc_signoff.TEXT",
    "ME_MOBILE.sfc_completed.TEXT",
    "OPERATION_BROWSE.title.TEXT",
    "RESOURCE_BROWSE.title.TEXT"
];
this.oLocaleTextValues = [];
this.localeErrorKeys = [
    "12903.simple",
    "12904.simple",
    "12906.simple",
    "12213.simple"
];
this.oLocaleErrorValues = [];

// requires JSON. if not defined do it here
if (!window.JSON) {
    loadJS("../scripts/json2.min.js");
}

// function called after Page has been loaded
function afterShow() {

    // If scripting is not allowed, this will cause the No Script message to display
    document.getElementById("noscriptmsg").style.display = "none";
    document.getElementById("pagecontainer").style.display = "block";

    // loads the appropriate CSS for the size specified in query parameter
    loadCSS(this.sCssPath + "style.css");
    loadCSS(this.sCssPath + "browse.css");
    loadCSS(this.sCssPath + "message.css");
    loadCSS("../lean/css/base/message_passive.css");
    document.getElementById("headerIcon").src=this.sCssPath + "img/info.png";
    document.getElementById("operationBrowseIcon").src=this.sCssPath + "img/action.png";
    document.getElementById("resourceBrowseIcon").src=this.sCssPath + "img/action.png";

    // change site first so we get correct operation / resource data
    try {
        if (!isBlank(this.sSite)) {
            window.sCurrentSite = this.sSite;
            initialize();
        } else {
            // Find the site for the current user. This will be the default site. If there is no default,
            // the first site the user is defined within is obtained.
            odataGetRequest("/manufacturing-odata/AppConfig.svc/GetSiteForUser", true, this.initialize, this);
        }
    }
}
```

```
} catch (err) {
    writeToLog("afterShow: " + err.message);
}
};

// function called directly if site passed in query parameter or
// as a callback from getting site from server
function initialize(oData) {

    // load site for current user
    if (oData && oData.GetSiteForUser) {
        window.sCurrentSite = oData.GetSiteForUser.site;
    }

    // if "&LOCALIZED_TEXT=true" query paramter is used, need to get text from server
    if (isBlank(this.sLocalizedText) || this.sLocalizedText.toLowerCase() === "true") {
        initializeLocalizedStandardText();

    // else initialize static text and setup view
    } else {
        initializeText();
        initializeView();
    }
};

// initialize standard text. this is an asynchronous request so results are passed
// to "initializeLocalizedErrorText()" callback function
function initializeLocalizedStandardText() {
    loadLocalizedValues("INFO", this.localeTextKeys, initializeLocalizedErrorText, this);
};

// initialize error text. this is an asynchronous request so results are passed
// to "initializeView()" callback function
function initializeLocalizedErrorText(aTextValues) {
    this.oLocaleTextValues = aTextValues;
    loadLocalizedValues("ERROR", this.localeErrorKeys, initializeView, this);
};

// initialize all text using hard-coded strings ("&LOCALIZED_TEXT=false" or not used)
function initializeText() {
    this.oLocaleTextValues["WIP_PROCESSING.startButton.LABEL"] = "Start";
    this.oLocaleTextValues["WIP_PROCESSING.signoffButton.LABEL"] = "Signoff";
    this.oLocaleTextValues["WIP_PROCESSING.completeButton.LABEL"] = "Complete";
    this.oLocaleTextValues["OPERATION_BROWSE.default.LABEL"] = "Operation";
    this.oLocaleTextValues["RESOURCE_BROWSE.default.LABEL"] = "Resource";
    this.oLocaleTextValues["SFC_BROWSE.default.LABEL"] = "SFC";
    this.oLocaleTextValues["error.title.TEXT"] = "Error";
    this.oLocaleTextValues["message.default.LABEL"] = "Message";
    this.oLocaleTextValues["ME_MOBILE.sfc_started.TEXT"] = "SFCs Started";
    this.oLocaleTextValues["ME_MOBILE.sfc_signoff.TEXT"] = "SFCs Signed Off";
    this.oLocaleTextValues["ME_MOBILE.sfc_completed.TEXT"] = "SFCs Completed";
    this.oLocaleTextValues["OPERATION_BROWSE.title.TEXT"] = "Operation Browse";
    this.oLocaleTextValues["RESOURCE_BROWSE.title.TEXT"] = "Resource Browse";

    this.oLocaleErrorValues["12903.simple"] = "Operation is missing (Message 12903)";
    this.oLocaleErrorValues["12904.simple"] = "Resource is missing (Message 12904)";
    this.oLocaleErrorValues["12906.simple"] = "SFC is missing (Message 12906)";
    this.oLocaleErrorValues["12213.simple"] = "No records found";
}

// called to initialize the page view (set labels and define event handlers
function initializeView(aTextValues) {
```

```
// if not undefined, then this is the error text
if (aTextValues) {
    this.oLocaleErrorValues = aTextValues;
}

var thisController = this;

// set up event handler to display information about the browser
var oElement = document.getElementById("headerIcon");
if (oElement) {
    addListener(oElement, "click", function(e) {
        thisController.showBrowserInformation();
    });
}

try {

    var oButton = document.getElementById("startButton");
    if (oButton) {
        changeElementText(oButton, this.oLocaleTextValues["WIP_PROCESSING.startButton.LABEL"]);
        addListener(oButton, "click", function(e) { thisController.start(); });
    }
    oButton = document.getElementById("signoffButton");
    if (oButton) {
        changeElementText(oButton, this.oLocaleTextValues["WIP_PROCESSING.signoffButton.LABEL"]);
        addListener(oButton, "click", function(e) { thisController.signoff(); });
    }
    oButton = document.getElementById("completeButton");
    if (oButton) {
        changeElementText(oButton, this.oLocaleTextValues["WIP_PROCESSING.completeButton.LABEL"]);
        addListener(oButton, "click", function(e) { thisController.complete(); });
    }

    changeElementTextById("operationLabel", this.oLocaleTextValues["OPERATION_BROWSE.default.LABEL"]);
    changeElementTextById("resourceLabel", this.oLocaleTextValues["RESOURCE_BROWSE.default.LABEL"]);
    changeElementTextById("sfcLabel", this.oLocaleTextValues["SFC_BROWSE.default.LABEL"]);

    oButton = document.getElementById("operationBrowseButton");
    if (oButton) {
        addListener(oButton, "click", function(e) { thisController.browseOperation(); });
    }

    oButton = document.getElementById("resourceBrowseButton");
    if (oButton) {
        addListener(oButton, "click", function(e) { thisController.browseResource(); });
    }

} catch (err) {
    writeToLog("afterShow: " + err.message);
    showMessageBox(err.message, this.oLocaleTextValues["error.title.TEXT"]);
}

this.sValidationErrorMessage = "";
if (this.sDefaultOperation) {
    validateOperation();
    return;
}

if (this.sDefaultResource) {
    validateResource();
    return;
}

setTimeout(this.adjustControls, 250);
};
```

// validates the query parameter Operation

```
function validateOperation() {
  var parameters="?operation=" + this.sDefaultOperation.toUpperCase() + "";
  var url = "/manufacturing-odata/Production.svc/ValidateOperation" + parameters;
  var thisObject = this;
  odataGetRequest(url, true,
    function(oData) {
      document.getElementById("operationField").value = thisObject.sDefaultOperation;
      if (thisObject.sDefaultResource) {
        thisObject.validateResource();
      } else {
        setTimeout(thisObject.adjustControls, 250);
      }
    },
    function(oError) {
      if (oError) {
        if (!isBlank(thisObject.sValidationErrorMessage)) {
          thisObject.sValidationErrorMessage = thisObject.sValidationErrorMessage + "<br/>";
        }
        thisObject.sValidationErrorMessage = thisObject.sValidationErrorMessage + oError.errorMessage;
      }
      if (thisObject.sDefaultResource) {
        thisObject.validateResource();
      } else {
        setTimeout(thisObject.adjustControls, 250);
      }
    },this);
}
```

// validates the query parameter Resource

```
function validateResource() {
  var parameters="?resource=" + this.sDefaultResource.toUpperCase() + "";
  var url = "/manufacturing-odata/Production.svc/ValidateResource" + parameters;
  var thisObject = this;
  odataGetRequest(url, true,
    function(oData) {
      document.getElementById("resourceField").value = thisObject.sDefaultResource;
      setTimeout(thisObject.adjustControls, 250);
    },
    function(oError) {
      if (oError) {
        if (!isBlank(thisObject.sValidationErrorMessage)) {
          thisObject.sValidationErrorMessage = thisObject.sValidationErrorMessage + "<br/>";
        }
        thisObject.sValidationErrorMessage = thisObject.sValidationErrorMessage + oError.errorMessage;
      }
      setTimeout(thisObject.adjustControls, 250);
    },this);
}
```

// adjust toolbar to be positioned below input fields

```
function adjustControls() {

  // force toolbar under the input fields
  var oDomInputContainer = document.getElementById("inputcontainer");
  if (oDomInputContainer) {
    var offsetTop = oDomInputContainer.offsetTop;
    var offsetHeight = oDomInputContainer.offsetHeight;
    var oToolbar = document.getElementById("toolbar");
    if (oToolbar) {
      oToolbar.style.top = offsetTop + offsetHeight + 10; // 10 for padding
    }
  }
}
```

```
// display any validation errors
if (!isBlank(this.sValidationErrorMessage)) {
    writeToLog(this.sValidationErrorMessage);
    showMessageBox(this.sValidationErrorMessage, this.oLocaleTextValues["error.title.TEXT"]);
    this.sValidationErrorMessage = undefined;
}
};

// function will display browser information (debugging purposes)
function showBrowserInformation() {
    var windowWidth = window.screen.width < window.outerWidth ? window.screen.width : window.outerWidth;
    var windowHeight = window.screen.height < window.outerHeight ? window.screen.height : window.outerHeight;
    if (!windowWidth || !windowHeight) {
        if (document.body && document.body.offsetWidth) {
            windowWidth = document.body.offsetWidth;
            windowHeight = document.body.offsetHeight;
        } else if (document.compatMode=="CSS1Compat" && document.documentElement.offsetWidth) {
            windowWidth = document.documentElement.offsetWidth;
            windowHeight = document.documentElement.offsetHeight;
        }
    }
    var message = "Width=" + windowWidth + ", Height =" + windowHeight + "<br>";
    message = message + "User Agent=" + navigator.userAgent + "<br>";
    showMessageBox(message, this.oLocaleTextValues["message.default.LABEL"]);
};

// function called to start an SFC
function start() {
    var requestData = this.getRequestData();
    if (!requestData) {
        return;
    }
    var parameters = "?Sfc=" + requestData.sfc + "&Operation=" + requestData.operation + "&Resource=" +
requestData.resource + "";
    var url = "/manufacturing-odata/Production.svc/StartSfc";
    odataRequest(url + parameters, "POST", null, this.startSuccessCallback, this.errorCallback, this);
};

// function called to signoff an SFC
function signoff() {
    var requestData = this.getRequestData();
    if (!requestData) {
        return;
    }
    var parameters = "?Sfc=" + requestData.sfc + "&Operation=" + requestData.operation + "&Resource=" +
requestData.resource + "";
    var url = "/manufacturing-odata/Production.svc/SignoffSfc";
    odataRequest(url + parameters, "POST", null, this.signoffSuccessCallback, this.errorCallback, this);
};

// function called to complete an SFC
function complete() {
    var requestData = this.getRequestData();
    if (!requestData) {
        return;
    }
    var parameters = "?Sfc=" + requestData.sfc + "&Operation=" + requestData.operation + "&Resource=" +
requestData.resource + "";
    var url = "/manufacturing-odata/Production.svc/CompleteSfc";
    odataRequest(url + parameters, "POST", null, this.completeSuccessCallback, this.errorCallback, this);
};

// helper function to get operation, resource and sfc
```

```
function getRequestData() {

    var operation = document.getElementById("operationField").value;
    if (!operation) {
        var message = this.oLocaleErrorValues["12903.simple"];
        showMessageBox(message, this.oLocaleTextValues["error.title.TEXT"]);
        return undefined;
    }
    var resource = document.getElementById("resourceField").value;
    if (!resource) {
        var message = this.oLocaleErrorValues["12904.simple"];
        showMessageBox(message, this.oLocaleTextValues["error.title.TEXT"]);
        return undefined;
    }
    var sfc = document.getElementById("sfcField").value;
    if (!sfc) {
        var message = this.oLocaleErrorValues["12906.simple"];
        showMessageBox(message, this.oLocaleTextValues["error.title.TEXT"]);
        return undefined;
    }

    var requestData = {
        operation : operation.toUpperCase(),
        resource : resource.toUpperCase(),
        sfc : sfc.toUpperCase()
    };

    return requestData;
};
```

// function called for successful Start

```
function startSuccessCallback(oData) {
    var sMessage = this.oLocaleTextValues["ME_MOBILE.sfc_started.TEXT"];
    showMessageBox(sMessage, this.oLocaleTextValues["message.default.LABEL"]);
};
```

// function called for successful Signoff

```
function signoffSuccessCallback(oData) {
    var sMessage = this.oLocaleTextValues["ME_MOBILE.sfc_signoff.TEXT"];
    showMessageBox(sMessage, this.oLocaleTextValues["message.default.LABEL"]);
};
```

// function called for successful Complete

```
function completeSuccessCallback(oData) {
    var sMessage = this.oLocaleTextValues["ME_MOBILE.sfc_completed.TEXT"];
    showMessageBox(sMessage, this.oLocaleTextValues["message.default.LABEL"]);
};
```

// function called if error occurred in Start, Signoff or Complete

```
function errorCallback(errorCode, errorMessage) {
    if (errorMessage) {
        writeToLog("WipProcessing Error: " + errorMessage);
        showMessageBox(errorMessage, this.oLocaleTextValues["error.title.TEXT"]);
    }
};
```

// functions called to display operation browse dialog

```
function browseOperation() {

    var url = "/manufacturing-odata/Production.svc/Operations?$skip=0&$top=100&$inlinecount=allpages&$format=json";
    var sOperation = document.getElementById("operationField").value;
    if (!isBlank(sOperation)) {
        url = url + "&$filter=operation%20eq%20'" + sOperation + "'";
    }
}
```

```
try {
  odataGetRequest(url, false,
    function(oData) {
      showBrowseOperationDialog(oData, sOperation);
    },
    function(errorCode, errorMessage) {
      writeToLog(errorMessage);
      showMessageBox(errorMessage, this.oLocaleTextValues["error.title.TEXT"]);
    },
    this);
} catch (err) {
  writeToLog(err.message);
  showMessageBox(err.message, this.oLocaleTextValues["error.title.TEXT"]);
  return;
}
};

function showBrowseOperationDialog(oData, sOperation) {
  if (!oData || oData.length === 0) {
    // 12213.simple=No records found
    var sMessage = this.oLocaleErrorValues["12213.simple"];
    writeToLog("browseOperation Error: " + sMessage + " for operation '" + sOperation + "'");
    showMessageBox(sMessage, this.oLocaleTextValues["error.title.TEXT"]);
    return;
  }

  var oProperties = [];
  oProperties[0] = "operation";
  oProperties[1] = "i18NDescription";
  browse(oData, oProperties, this.oLocaleTextValues["OPERATION_BROWSE.title.TEXT"],
    this.sSizeUI, this.operationBrowseCallback, this);
};
```

// function called with operation selection

```
function operationBrowseCallback(oData) {
  if (oData) {
    document.getElementById("operationField").value = oData.operation;
    closeBrowse();
  }
};
```

// functions called to display resource browse dialog

```
function browseResource() {

  var url = "/manufacturing-odata/Production.svc/Resources?$skip=0&$top=100&$inlinecount=allpages&$format=json";

  var sResource = document.getElementById("resourceField").value;
  if (!isBlank(sResource)) {
    url = url + "&$filter=resource%20eq%20%27" + sResource + "%27";
  }
  try {
    odataGetRequest(url, false,
      function(oData) {
        showBrowseResourceDialog(oData, sResource);
      },
      function(errorCode, errorMessage) {
        writeToLog(errorMessage);
        showMessageBox(errorMessage, this.oLocaleTextValues["error.title.TEXT"]);
      },
      this);
  } catch (err) {
    writeToLog(err.message);
    showMessageBox(err.message, this.oLocaleTextValues["error.title.TEXT"]);
    return;
  }
}
```

```
};
function showBrowseResourceDialog(oData, sResource) {
  if (!oData || oData.length === 0) {
    // 12213.simple=No records found
    var sMessage = this.oLocaleErrorValues["12213.simple"];
    writeToLog("browseOperation Error: " + sMessage + " for resource '" + sResource + "'");
    showMessageBox(sMessage, this.oLocaleTextValues["error.title.TEXT"]);
    return;
  }

  var oProperties = [];
  oProperties[0] = "resource";
  oProperties[1] = "i18NDescription";
  browse(oData, oProperties, this.oLocaleTextValues["RESOURCE_BROWSE.title.TEXT"],
    this.sSizeUI, this.resourceBrowseCallback, this);
};

// function called with resource selection
function resourceBrowseCallback(oData) {
  if (oData) {
    document.getElementById("resourceField").value = oData.resource;
    closeBrowse();
  }
};

</script>

</head>

<body id="content" onLoad="afterShow()">

  <div id="pagecontainer" class="pageContainer">

    <div id="headercontainer" class="headerRow">
      <label id="headerLabel" class="headerText">WIP Processing</label>
      <img id="headerIcon" class="headerInfoIcon"></img>
    </div>

    <div id="inputcontainer" class="inputControlRows">

      <div class="inputControlFields">
        <div class="leftColumn">
          <div class="labelContainer">
            <label id="operationLabel" for="operationField" class="truncatedLabel fieldLabel"></label>
          </div>
        </div>
        <div class="rightColumn">
          <div class="browseContainer">
            <div class="browseFieldContainer">
              <input id="operationField" class="inputUpperCase operationField" type="text" ></input>
            </div>
            <div class="browseButtonContainer">
              <button id="operationBrowseButton" class="browseButton" type="button">
                <img id="operationBrowseIcon" class="browseButtonIcon"></img>
              </button>
            </div>
          </div>
        </div>
      </div>
      <div class="leftColumn">
        <div class="labelContainer">
          <label id="resourceLabel" for="resourceField" class="truncatedLabel fieldLabel"></label>
        </div>
      </div>
      <div class="rightColumn">
        <div class="browseContainer">
```

```
<div class="browseFieldContainer">
  <input id="resourceField" class="inputUpperCase resourceField" type="text"></input>
</div>
<div class="browseButtonContainer">
  <button id="resourceBrowseButton" class="browseButton" type="button">
    <img id="resourceBrowseIcon" class="browseButtonIcon"></img>
  </button>
</div>
</div>
<div class="leftColumn">
  <div class="labelContainer">
    <label id="sfcLabel" for="sfcField" class="truncatedLabel fieldLabel"></label>
  </div>
</div>
<div class="rightColumn">
  <input id="sfcField" class="inputUpperCase sfcField" type="text" ></input>
</div>
</div>
<div id="toolbar" class="toolbarRow">
  <button id="startButton" class="toolbarButton" type="button"></button>
  <button id="signoffButton" class="toolbarButton" type="button"></button>
  <button id="completeButton" class="toolbarButton" type="button"></button>
</div>
<div id="noscriptmsg" class="noScriptMessage">
JavaScript is not enabled and is required to run WIP.
</div>
</body>
</html>
```

5.1 Ultra-Lean JavaScript functions

The example above uses utility functions defined in `ultralean.js`. The following documentation describes the usage for the available functions.

```
/**
 * Will perform an asynchronous request using the input servletUrl.
 *
 * @param servletUrl url of servlet
 * @param requestType Must be either "POST" or "GET"
 * @param requestData JSON object to post in request
 * @param successCallback function called with response JSON data:
 * <pre>
 *   callback(oData)
 * </pre>
 * @param errorCallback function called when error occurs:
 * <pre>
 *   callback(errorCode, errorMessage)
 * </pre>
 * @param fnContext context to call success and error functions with
 * @throws Error if request returns exception
 */
odataRequest = function(servletUrl, requestType, requestData, successCallback, errorCallback, fnContext);

/**
 * Will perform an asynchronous request to an odata service.
 *
 * @param sPath path to odata service
 * @param bGetAllData true to get all data including meta data, false to get "results" portion of response only
 * @param fnCallback function called with response JSON data:
 * <pre>
```

```
* callback(oData)
* </pre>
* @param fnErrorCallback function called when error occurs:
* <pre>
*   callback(errorCode, errorMessage)
* </pre>
* @param fnContext context to call success and error functions with
* @throws Error if request returns exception
*/
odataGetRequest = function(sPath, bGetAllData, fnCallback, fnErrorCallback, fnContext);

/**
 * Performs an asynchronous request to get a locale-specific string value for the given key sKey.
 *
 * If text parameters are given, then any occurrences of the pattern "{<i>n</i>}" with <i>n</i> being an integer
 * are replaced by the parameter value with index <i>n</i>. Note: This replacement is also applied if no text had been found (key).
 *
 * @param sKey string key in bundle to get value for
 * @param oProperties Map parameters which should replace the place holders "%..." tokens
 *               in the found locale-specific string value. Example: oProperties["KEY"] = value;
 * @param fnCallback function called with the key and value
 * <pre>
 *   callback(sKey, sValue)
 * </pre>
 * @param fnContext context to call success and error functions with
 */
localeSpecificText = function(sKey, oProperties, fnCallback, fnContext);

/**
 * Performs an asynchronous request to get a locale-specific string value for the given error key sKey.
 *
 * If text parameters are given, then any occurrences of the pattern "%<i>s</i>%" with <i>s</i> being a string value
 * are replaced by the parameter value with key "%<i>s</i>". Note: This replacement is also applied if no text had been found (key).
 *
 * @param sKey string key in bundle to get value for
 * @param oProperties Map parameters which should replace the place holders "%..." tokens
 *               in the found locale-specific string value. Example: oProperties["KEY"] = value;
 * @param fnCallback function called with the key and value
 * <pre>
 *   callback(sKey, sValue)
 * </pre>
 * @param fnContext context to call success and error functions with
 */
errorText = function(sKey, oProperties, fnCallback, fnContext);

/**
 * Performs an asynchronous request to get a locale-specific string value for the given key sKey.
 *
 * If text parameters are given, then any occurrences of the pattern "%<i>s</i>%" with <i>s</i> being a string value
 * are replaced by the parameter value with key "%<i>s</i>". Note: This replacement is also applied if no text had been found (key).
 *
 * @param sMsgType string type of text to get (eg; "ERROR" or "INFO")
 * @param sKey string key in bundle to get value for
 * @param oProperties Map parameters which should replace the place holders "%..." tokens
 *               in the found locale-specific string value. Example: oProperties["KEY"] = value;
 * @param fnCallback function called with the key and value
 * <pre>
 *   callback(sKey, sValue)
 * </pre>
 * @param fnContext context to call success and error functions with
 */
i18nTextRequest = function(sMsgType, sKey, oProperties, fnCallback, fnContext);

/**
 * Performs an asynchronous request to load a list of locale-specific string values from a list of keys
 *
```

```
* @param sMsgType string type of text to get (eg; "ERROR" or "INFO")
* @param aTextKeys array of keys in bundle to get values for
* @param fnCallback function called after all values are loaded. It is passed an
*       array defined as aTextValues["key"] = value
* <pre>
*   callback(aTextValues)
* </pre>
* @param fnContext context to call functions with
*/
loadLocalizedValues = function(sMsgType, aTextKeys, fnCallback, fnContext);

/**
* loads a given CSS file
*
* @param sURL string Uniform resource locator of CSS file to load.
*/
loadCSS = function(sURL);

/**
* loads a given JavaScript file
*
* @param sURL string Uniform resource locator of JavaScript file to load.
*/
loadJS = function(sURL) {};

/**
* helper function to see if input string is blank
*
* @param str String to check
* @return true if string is blank (undefined, null or only spaces), else true
*/
isBlank = function(str);

/**
* helper function to trim leading and trailing spaces from a string
*
* @param str String to trim spaces from
* @return updated String
*/
trimString = function(str);

/**
* helper function to encode the input string (including single quotes)
*
* @param str String to encode
* @return encoded string
*/
encodeString = function(str);

/**
* helper function to check if one string ends with another
*
* @param str String to check
* @param sSearch String to search
* @return true if str ends with sSearch
*/
endsWith = function(str, sSearch);

/**
* This utility method substitutes in the String _orig all occurrences of string
* _to substr with the string _withsubst
*
* @param _orig The string to search
* @param _to substr what to search for
* @param _withsubst what to substitute with
* @param _startIndex start replacing from this index
```

```
*
* @return String new string with substitution executed
*/
replaceStrings = function(_orig, _tosubst, _withsubst, _startIndex);

/**
* writes string to debugger console if debugger is loaded
*
* @param str string to write to console
*/
writeToLog = function(str);

/**
* cross-browser function to add an event listener to a DOM element
*
* @param oElement DOM element to add listener for
* @param sEvent event to listen for (i.e.; "click", "resize", etc)
* @param fHandler event handler function
*/
addListener = function(oElement, sEvent, fHandler);

/**
* cross-browser function to remove an event listener from a DOM element
*
* @param oElement DOM element to remove listener from
* @param sEvent event to remove handler for (i.e.; "click", "resize", etc)
* @param fHandler event handler function
*/
removeListener = function(oElement, sEvent, fHandler);

/**
* displays a floating dialog box with the input message and title
*
* @param sMessage message to display
* @param sTitle title of dialog
*/
showMessageBox = function(sMessage, sTitle);

/**
* displays a floating dialog box with the input content
*
* @param oContent DOM elements to show in dialog
* @param sDialogBkgColor Optional background color
* @param oDialogCloseIcon Optional IMG element to show for close icon
* @param fnAfterShow Optional function to call after dialog is shown
* @param oFnContext Optional context to call function with
*/
showDialogBox = function(oContent, sDialogBkgColor, oDialogCloseIcon, fnAfterShow, oFnContext);

/**
* Closes the floating dialog box rendered using showDialogBox().
*/
closeDialogBox = function();

/**
* displays a "browse" dialog with the input
*
* @param aData Object data to return based on selection
* @param aDisplayProperties[0-1] Display text for each data object
* @param sTitle Title for browse dialog
* @param sSizeUI Size for dialog ("LARGE", "MEDIUM" or "SMALL")
* @param fnCallback callback function to call with selection (i.e.; fnCallback(aData[selection index])
* @param oFnContext Optional context to call function with
*/
browse = function(aData, aDisplayProperties, sTitle, sSizeUI, fnCallback, oFnContext);
```

```
/**
 * Closes the floating dialog box rendered using browse().
 */
closeBrowse = function();

/**
 * cross browser function to change text of an element by id
 *
 * @param sElement DOM element id
 * @param sText text
 */
changeElementTextById = function(sElementId, sText);

/**
 * cross browser function to change text of an element
 *
 * @param oElement DOM element
 * @param sText text
 */
changeElementText = function(oElement, sText);

/**
 * Converts a String in JSON format to a JSON object
 *
 * @param sData String in JSON format
 * @return a JSON object
 */
getJSONObject = function(sData);

/**
 * Stringifies a given object from a JSON object
 * @param oData a non-recursive JSON object
 * @param sFormat Formatting option TEXT,HTML for screen output (optional)
 * @returns The string representation of the object.
 */
stringifyJsonObject = function( oData, sFormat );
```