Intelligent Offer Management (IOM),
Version 2.3.1

IOM Legacy JSP ASP Reference

NUANCE

Notice

Intelligent Offer Management (IOM) Version 2.3.1
IOM Legacy JSP ASP Reference

1198 E. Arques Avenue, Sunnyvale, CA 94085, U.S.A.
Printed in the United States of America.

Last updated 3/5/08.

Contents

CHAPTER 1 OVERVIEW TO IOM LEGACY JSP API 5
IOM Documentation 5

CHAPTER 2 JSP API 7
Equivalences of HTTP API and Legacy JavaScript API 7
Developer Touchpoints with the JSP API 7
Adding Slots to a Voice Application 7
Customizing initialize.jsp for Your Own Demographic Info 8
Destination Maps and iom_destinations.vxml.jsp 9
Self-Fulfillment in iom_promo.vxml.jsp 10
JSP Reference 10
Location of Installed JSPs 10
Summary of JSP API 11
Typical Calling Sequence 11
create.jsp 12
destroy.jsp 13
initialize.jsp 14
iom_promo.vxml.jsp 16
report_purchase.jsp 18
update.jsp 19
Sample IOM-based Voice Application 20

Chapter 1 Overview to IOM Legacy JSP API

Note: This guide documents the Version 1.6.x Java Server Pages (JSP) Application Programming Interface (API) for Intelligent Offer Management (IOM), which is considered legacy.

You are encouraged to use the latest HTTP API, instead. See the *IOM Developer and Datafeed Guide*.

IOM Documentation

Intelligent Offer Management (IOM) is documented in the following books.

Book Title	Audience	Description
IOM Concepts Guide	All readers	■ Overview concepts for IOM: □ Theory of general operation □ Components of the IOM system □ Definitions of key terminology ■ Tutorials of campaigns from start to finish: □ Campaign design □ Implementing campaign objects in the IOM Tool □ Voice application design with a sample application ■ Release notes for latest version of IOM
IOM Tool User Guide	Campaign administrators	Step-by-step use of the IOM Tool for all tasks: ■ Managing campaign objects ■ Importing and exporting applications, campaigns, or offers ■ IOM user set-up in C3
IOM Developer and Datafeed Reference	Voice application developers	All details relating to programming IOM-based voice applications and IOM datafeed: ■ Theory of programming, development methodology ■ XML-over-HTTP API ■ Developing and testing an IOM datafeed, alarms, and service monitors.

Book Title	Audience	Description
IOM Legacy JSP ASP Reference	Developers supporting IOM 1.6 legacy applications	Details related to the pre-IOM-2.3 legacy Java Server Pages Application Programming Interface.

Chapter 2 JSP API

Equivalences of HTTP API and Legacy JavaScript API

The following table maps the legacy JavaScript API functions to their equivalents in the HTTP API.

Table 1 Legacy JSP (IOM Version 1.6) and HTTP API Equivalents

Legacy JSP	HTTP API
`create.jsp`	`create.do`
`initialize.jsp`	`update.do`
`PromotionPlayer.playCurrentPromotion`	`getPromotion.do`
`update.jsp`	`reportPromotion.do`
`report_purchase.jsp`	`reportPurchase.do`
`destroy.jsp`	`destroy.do`

Developer Touchpoints with the JSP API

This section describes some of the customization points in coding a voice application for IOM withe Java Server Pages API. For background information about the JSPs, see “Summary of JSP API” on page 11.

Adding Slots to a Voice Application

You must add at least one slot to your IOM-enabled voice application. Slot names must match the names you create with the IOM Tool.

To add a slot to your voice application, pass the slot name to `iom_promo.vxml.jsp` as shown in the following example. The `slotName` passed to `iom_promo.vxml.jsp` exactly matches the name of a slot created with the IOM Tool: in this example, `'At Top Menu 2'`.

```
<var name="applicationName"
   expr="'<bv:ConfigValuekey="promo.applicationName"/>'"/>
<var name="rsp" expr="'<%=request.getParameter("rsp")%>'"/>
<var name="locale"
   expr="'<%=request.getParameter("locale")%>'"/>
<var name="slotName" expr="'At Top Menu 2'"/>
<subdialog name="promo1"
   src="<bv:ConfigValue key="iom_play_url"/>"
   namelist="bevocal.sessionid providerID rsp locale
   slotName">
</subdialog>
```

For an explanation of the `iom_promo.vxml.jsp` arguments, see Table 6, "iom_promo.vxml.jsp Input Parameters," on page 16.

Customizing initialize.jsp for Your Own Demographic Info

Customizing the behavior of `initialize.jsp` is optional.

If your voice application itself collects demographic information for use by IOM, you must modify the following statement in `initialize.jsp` to pass the `DemographicInfo` object instead of `null` on the `initializePromotionPlayer` method:

```
PromotionPlayer.initializePromotionPlayer(sessionId,
providerID, min, null);
```

Your voice application must supply an implementation class by using the `DemographicInfo` Java interface. The `DemographicInfo` interface provides data access for the demographic variables in real time (most likely, the carrier's back-end hits). The data values drive the demographic rules used by IOM promotions. The `DemographicInfoItem` key must match the names of segment variables defined with the IOM Tool.

The status of the `DemographicInfoItem` object returned by `getDemographicInfoItem()` *must* be `SUCCESS` if the object is to be used by IOM. Otherwise, all rules based on this demographic information evaluate to `false`.

Destination Maps and iom_destinations.vxml.jsp

Destination mapping is a mechanism for IOM to control a voice application's logical branching based on a subscriber's response to an interactive promotion. Configuring a destination map is optional. You may want your voice application to completely control logical branching without assistance from IOM.

A destination map is a file you upload for a specific promotion with the IOM Tool (see the *IOM Tool User Guide*). For each promotion, the destination map is composed of key/value pairs of grammar response values and the corresponding forms subdialogs to which IOM should transfer after receiving them. The following is a simple `config.xml` destination map with only a single key:

```
<?xml version="1.0"?>
<config>
   <item key="yes" value="Purchase_Feature_Dialog" />
</config>
```

Note: The mapped subdialogs (the values in your destination map) *must* be in the file `iom_destinations.vxml.jsp`, which you must create yourself. The location of this file is based on the RSP path for your application.

The `iom_destinations.vxml.jsp` is invoked automatically by `iom_promo.vxml.jsp`; you do not have to call it, but you must create it. The page can do whatever it must to complete the fulfillment, such as making back-end calls, querying the database, or branching to other subdialogs, but it *must* do the following:

- Invoke `report_purchase.jsp` to record the purchase
- Return control to IOM after it completes

For an example of an `iom_destinations.vxml.jsp`, see "Sample IOM-based Voice Application" on page 20.

In some implementations, such as the AT&T *NOW, IOM is embedded within the Voice Store voice application itself, so there is no easy place to transfer from IOM to fulfill the offer. Hence, this implementation does not use destination mapping. Instead, IOM simply returns control to the voice application after accepting the response from the caller, and the voice application itself interprets the response to perform fulfillment. In this case, a dummy destination map file with an empty key is still required, such as the following:

```
<?xml version="1.0"?>
<config>
```

```
    <item key="yes" value="" />
</config>
```

Because no destination is specified, `iom_promo.vxml.jsp` returns control to the voice application, which must invoke `report_purchase.jsp` to record the purchase.

Self-Fulfillment in iom_promo.vxml.jsp

If your voice application does its own fulfillment, do the following check after the return from `iom_promo.vxml.jsp`.

```
if promoType == 'interactive' &&
   productCode != '' &&
   destinationKey != 'no'
then
{
>> the calling application should fulfill the promotion <<
   }
```

JSP Reference

This section presents the formal syntax and examples of the Intelligent Offer Management (IOM) Java Server Pages API.

Location of Installed JSPs

The absolute path to the JSPs depends on where the JSPs have been installed. In this section, the formal syntax shows this location as:

`$`*`jspLocation`*

and the examples show this location as:

`/services/iom_interface/`*`jspName`*

Summary of JSP API

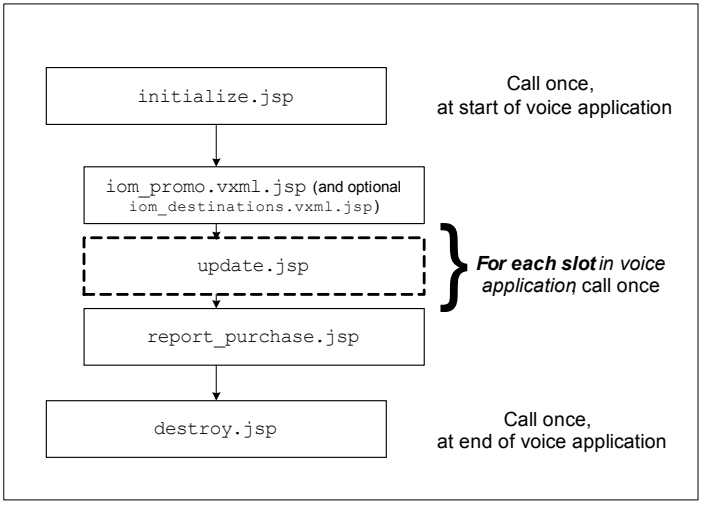
The following is a summary of the IOM Java Server Pages for use by your voice application JSP.

Table 2 IOM Java Server Pages

JSP	Description
`create.jsp`	Deprecated. Use `initialize.jsp` instead.
`destroy.jsp`	At the end of the session, writes IOM information to the database and destroys the PromotionPlayer object.
`intialize.jsp`	Initializes the PromotionPlayer object.
`iom_promo.vxml.jsp`	Plays the promotion for the current slot. This JSP must be called for each slot in the voice application. Optionally, this JSP relies on a destination map of subdialogs contained in `iom_destinations.vxml.jsp`, which you must create.
`report_purchase.jsp`	Reports each subscriber purchase via an IOM offer.
`update.jsp`	Records whether the subscriber heard a promotion and accepted it.

Typical Calling Sequence

The figure below illustrates the typical sequence in which you call the IOM JSPs. Optional JSPs are shown in dashed boxes.



create.jsp

Note: Deprecated. Do not use. Use `initialize.jsp` instead.

`create.jsp` must be invoked before other IOM JSPs to create the `PromotionPlayer` object.

Syntax:

```
<bv:Data name="createInfo"
   srcKey="$jspLocation/create.jsp"
   namelist="bevocal.sessionid providerID"
   assign="createStatus">
```

Table 3 create.jsp Input Parameters

Argument	Description
`bevocal.sessionid`	VXML session ID
providerID	The name of the customer's application as defined in C3

Example

```
<block>
<bv:Data name="createInfo"
       srcKey="$/services/iom_interface/create.jsp"
       namelist="bevocal.sessionid providerID"
       assign="createStatus">
   <if cond="createStatus != undefined &amp;&amp;
             createStatus.status == 'success'">
       <log>Successful IOM creation</log>
       <assign name="iomActive" expr="true"/>
       <bv:Goto to="#IOMInit" goback="false"/>
   <else/>
       <log>Failed IOM creation</log>
       <disconnect/>
   </if>
</bv:Data>
</block>
```

Responses

Success:

```
<?xml version=1.0 encoding="iso-8589-1"?>
<return>
```

```
   <item key="status">success</item>
<return>
```

Failure:

```
<?xml version=1.0 encoding="iso-8589-1"?>
<return>
   <item key="status">error</item>
</return>
```

destroy.jsp

`destroy.jsp` destroys the PromotionPlayer object and writes all information to the IOM database.

Syntax:

```
<bv:Data name="destroyInfo"
   srcKey="$jspLocation/destroy.jsp"
   namelist="bevocal.sessionid"
   assign="destroyStatus">
```

Table 4 destroy.jsp Input Parameters

Argument	Description
`bevocal.sessionid`	VXML session ID

Example

```
<block>
   <if cond="iomActive == true">
      <bv:Data name="destroyInfo"
            srcKey="$/services/iom_interface/destroy.jsp"
            namelist="bevocal.sessionid"
            assign="destroyStatus">
      <if cond="destroyStatus != undefined &amp;&amp;
         destroyStatus.status == 'success'">
         <log>Successful IOM destroy</log>
         <assign name="iomActive" expr="false"/>
      <else/>
         <log>Failed IOM destroy</log>
      </if>
      </bv:Data>
    </if>
</block>
```

Responses

Success:

```
<?xml version=1.0 encoding="iso-8589-1"?>
<return>
   <item key="status">success</item>
</return>
```

Failure:

```
<?xml version=1.0 encoding="iso-8589-1"?>
<return>
   <item key="status">error</item>
</return>
```

initialize.jsp

`initialize.jsp` creates the PromotionPlayer object and sets the Mobile Identification Number (MIN) and associated demographic information in it. The MIN and demographic information must be obtained by the voice application before invoking `initialize.jsp`.

For information about customizing initialize.jsp to pass your own demographic information, see "Customizing initialize.jsp for Your Own Demographic Info" on page 8.

Syntax:

```
<bv:Data name="initInfo"
      srcKey="$/services/iom_interface/initialize.jsp"
      namelist="bevocal.sessionid providerID min
         demographicInfo"
      assign="initStatus">
```

Table 5 initialize.jsp Input Parameters

Argument	Description
`bevocal.sessionid`	VXML session ID
providerID	The name of the customer's application as defined in C3
min	Mobile Identification Number

Table 5 initialize.jsp Input Parameters

Argument	Description
demographicInfo	Your own custom demographic information object associated with the MIN. For details about setting the DemographicInfo object, see "Customizing initialize.jsp for Your Own Demographic Info" on page 8. If set to null, no demographic information is passed.

Example

```
<bv:Form id="IOMInit">
<block>
<bv:Data name="initInfo"
       srcKey="$/services/iom_interface/initialize.jsp"
       namelist="bevocal.sessionid providerID min"
       assign="initStatus">
   <if cond="initStatus != undefined &amp;&amp;
       initStatus.status == 'success'">
       <log>Successful IOM initialization</log>
       <bv:Goto to="#Pause" goback="false"/>
   <else/>
       <log>Failed IOM initialization</log>
       <disconnect/>
   </if>
</bv:Data>
</block>
</bv:Form>
```

Responses

Success:

```
<?xml version=1.0 encoding="iso-8589-1"?>
<return>
   <item key="status">success</item>
</return>
```

Failure:

```
<?xml version=1.0 encoding="iso-8589-1"?>
<return>
   <item key="status">error</item>
</return>
```

iom_promo.vxml.jsp

`iom_promo.vxml.jsp` plays a promotion and must be called for each slot in the voice application.

Optionally, this JSP relies on the following files:

- A destination map in the file `config.xml` that you create and upload with the IOM Tool.
- A JSP called `iom_destinations.vxml.jsp` that you craft by hand (see "Destination Maps and iom_destinations.vxml.jsp" on page 9).

If the subscriber accepts the promotion, control is transferred to the subdialog in `iom_destinations.vxml.jsp`; `iom_promo.vxml.jsp` automatically passes some of its response fields to `iom_destinations.vxml.jsp`. Otherwise, if the destination map and `iom_destinations.vxml.jsp` do not exist, after `iom_promo.vxml.jsp` is called, control is returned to the calling voice application.

Syntax:

```
<var name="slotName" expr="bvFormId"/>
<subdialog name="iom_promo" src="<bv:ConfigValue
   key="$jspLocation/vxml/iom_promo.vxml.jsp"/>"
   namelist="bevocal.sessionid providerID rsp locale slotName">
</subdialog>
```

Table 6 iom_promo.vxml.jsp Input Parameters

Argument	Description
`bevocal.sessionid`	VXML session ID
providerID	The name of the customer's application as defined in C3
rsp	Resource service path. A standard static piece of code that provides resources for the slot to function properly.
locale	The language locale for the promotion
slotName	The name of the slot as defined with the IOM Tool

Example

```
<block>
    <var name="slotName" expr="bvFormId"/>
    <subdialog name="iom_promo" src="<bv:ConfigValue
    key="$/services/iom_interface/vxml/iom_promo.vxml.jsp"/>"
           namelist="bevocal.sessionid providerID
           rsp locale slotName">
       </subdialog>
    <log>Status: <value expr="iom_promo.status"/></log>
    <log>Desc: <value expr="iom_promo.desc"/></log>
    <log>ProductCode:
    <value expr="iom_promo.productCode"/>
    </log>
    <log>PromoType: <value expr="iom_promo.promoType"/></log>
    <log>DestinationKey:
    <value expr="iom_promo.destinationKey"/>
    </log>

    <goto next="#At_Top_Menu_2"/>
</block>
```

Responses

Table 7 Response Fields from iom_promo.vxml.jsp

Response Field	Description
`status`	Status code of the call. Possible values are: ■ `failure`: An exception occurred in `iom_promo.vxml.jsp`. ■ `opt_out`: The subscriber did not respond or there was no match. ■ `success`
`desc`	Description of the status
`productCode`	The promotion's product code specified when the promotion was created with the IOM Tool. Pass `productCode` to `report_purchase.jsp`. Note: This field is also passed to `iom_destinations.vxml.jsp`.
`promoType`	Possible values are: ■ `interactive` ■ `informational` Note: This field is also passed to `iom_destinations.vxml.jsp`.
`destinationKey`	Name of key defined in destination map (`config.xml`), if such a map was uploaded with the IOM Tool. The value of this key is a grammar subdialog that must be defined in `iom_destinations.vxml.jsp`. Note: This field is also passed to `iom_destinations.vxml.jsp`.

report_purchase.jsp

`report_purchase.jsp` reports whether the product or service in the promotion has been purchased by the subscriber or not.

Syntax:

```
<bv:Data name="purchaseInfo"
srcKey="$jspLocation/report_purchase.jsp"
   namelist="bevocal.sessionid providerID
   productCode isProductPurchased"
   assign="purchaseStatus">
```

Table 8 report_purchase.jsp Input Parameters

Argument	Description
`bevocal.sessionid`	VXML session ID
providerID	The name of the customer's application as defined in C3
productCode	Value returned from `iom_promo.vxml.jsp`. See Table 7, "Response Fields from iom_promo.vxml.jsp," on page 17
isProductPurchased	`true` or `false`

Example

```
<var name="productCode" expr="iom_promo.productCode"/>
var name="isProductPurchased" expr="'true'"/>
<if cond="iom_promo.promoType == 'interactive' &amp;&amp;
iom_promo.productCode != '' &amp;&amp;
iom_promo.destinationKey != 'no'">
   <log> fulfillment menu </log>
   <log> purchase report of iom </log>
   <bv:Data name="purchaseInfo"
   srcKey="$/services/iom_interface/report_purchase.jsp"
      namelist="bevocal.sessionid providerID
      productCode isProductPurchased"
      assign="purchaseStatus">
   <if cond="purchaseStatus != undefined &amp;&amp;
      purchaseStatus.status == 'success'">
      <log>Successful IOM purchase reporting</log>
   <else/>
      <log>Failed IOM purchase reporting</log>
```

```
   </if>
   </bv:Data>
</if>
```

Responses

Success:

```
<?xml version=1.0 encoding="iso-8589-1"?>
<return>
   <item key="status">success</item>
</return>
```

Failure:

```
<?xml version=1.0 encoding="iso-8589-1"?>
<return>
   <item key="status">error</item>
</return>
```

update.jsp

`update.jsp` records the subscriber's status: whether the subscriber listened to the promotion and accepted it.

Syntax:

```
<bv:Data name="updateInfo"
   srcKey="$jspLocation/update.jsp"
   namelist="bevocal.sessionid providerID promoId
accepted"
   assign="updateStatus">
```

Table 9 update.jsp Input Parameters

Argument	Description
`bevocal.sessionid`	VXML session ID
providerID	The name of the customer's application as defined in C3
promoId	The name of the promotion that was played to the subscriber
accepted	Either `true` or `false`

Responses

Success:

```
<?xml version=1.0 encoding="iso-8589-1"?>
<return>
   <item key="status">success</item>
</return>
```

Failure:

```
<?xml version=1.0 encoding="iso-8589-1"?>
<return>
   <item key="status">error</item>
</return>
```

Sample IOM-based Voice Application

Nuance provides a sample IOM-JSP-based voice application to help developers understand how to integrate IOM into a voice application.

The sample voice application:

- Plays a welcome prompt
- Initializes the IOM session
- Creates two IOM slots: `At_Top_Menu_1` and `At_Top_Menu_2`
- Uses a `config.xml` destination map with an example `iom_destinations.vxml.jsp`.
- Destroys the IOM session.
- Plays a good bye prompt.

The sample voice application is in the IOM installation at:

installDir`/demo`

Index

C

config.xml for destination file 9, 16, 20
create.jsp (deprecated) 12

D

DemographicInfo 14
  DemographicInfoItem key for name of variable 8
  getDemographicInfoItem 8
  interface 8
  object 8
demographicInfo 15
Destination file 9
  with iom_promo.vxml.jsp 16
destroy.jsp 13

G

getDemographicInfoItem Java class 8

I

initialize.jsp 12, 14
  customizing 8
initializePromotionPlayer Java method 8
IOM Tool
  for slots 7
iom_destinations.vxml.jsp 9, 16, 20
iom_promo.vxml.jsp 7, 8, 10, 16, 18
isProductPurchased 18

J

Java methods/interfaces
  DemographicInfo 8
  getDemographicInfoItem 8
  initializePromotionPlayer 8
JSP API
  create.jsp (deprecated) 12
  destroy.jsp 13
  initialize.jsp 12, 14
    customizing 8
  iom_promo.vxml.jsp 7, 8, 10, 16, 18
  report_purchase.jsp 9, 10, 18
  update.jsp 19
jspLocation 10

M

min 14
Mobile Identification Number 14

P

productCode 17, 18
PromotionPlayer
  class 8
  object 12

R

report_purchase.jsp 9, 10, 18
Resource service path 16
RSP. See Resource service path.

S

Sample voice application 20
services/iom_interface 10
sessionid 12, 13, 14, 16, 18, 19
slotName 16
Slots
    adding to voice application 7

U

update.jsp 19