

## The virtually unknown: Record/Playback feature in Oracle Applications 11i comes alive!

Glenn Bruns  
*BOSS Corporation*

Bill Dunham  
*BOSS Corporation*

### Introduction

This document is about a somewhat unknown feature in Oracle Applications 11i called Record/Playback. It has been around since the Character mode version of the Apps and is a feature rarely used. It allows one to record and playback keystrokes as they were recorded, over and over again. The steps are simple; edit two files, record the session, make changes to the newly recorded forms event file, and play it back. Performing these steps will automate the newly recorded changes. This paper will walk through each step in detail with helpful hints and precautions along the way.

To perform the changes introduced in this document, one should have a working knowledge of Application DBA skills, as well as an understanding of HTML code.

During the setup of the Record/Playback feature, certain files need to be edited. Make a backup of all these files before you get started. The backup will allow recovery if unknown mistakes are made while adding the Record/Playback feature to your applications.

### Editing the appsweb.cfg

Edit the \$OA\_HTML/bin/appsweb.cfg file so the Record/Playback feature will be recognized. This is the configuration file used by the Apache during the initial connection to spawn a forms session. Edit the Special Functionality Parameters section and add *play=* directly below the *record* parameter line. Save the changes. *Note:* Leave the *record* and *play* parameters blank. This value will be decided in the URL while accessing the applications. If a parameter is included after the equals sign, then the parameter will be the name of the file and the file will be recreated each and every time a user accesses the application. Below is an example of the correct entry:

```
; 7) Special Functionality Parameters
; -----
; Record parameter values include:
; - performance : records server events timings into log file
; - collect    : records Runtime Diagnostic data into log file
; - all        : records Diagnostic and Performance data
; - names      : adds UI names to messages, no log generated
; - pecs       : old performance data saved into log file
; Otherwise no recording takes place.
record=
;Play needs to be defined to allow playback
play=
;
; Log parameter specifies log file location and name. If no
; log value is specified the default is CollectionType_PID.log
log=
```

## Adding the play parameter in the Appbase.htm

For the Record/Playback feature to run, you must edit the appbase.htm file. This is the default HTML file needed to startup an applet. It points to the appweb.cfg for values needed while displaying the Application sign on form.

Before editing the \$OA\_HTML/US /appbase.htm file, check the appweb.cfg file too see if the *baseHTML* parameter points to the file appbase.htm. If not, make changes to the file it points to instead of the appbase.htm. Edit the designated file and include the *play* parameter under the Special Function Parameter section. See the example below:

```
// Special Function Parameters
// -----
var xrecord    = "%record%"
var xplay     = "%play%"
var xlog       = "%log%"
var xnetStats  = "%netStats%"
var xhtmldebug = "%htmlDebug%"
var xORBdL     = "%ORBdisableLocator%"
// Currently Not Used:
var xHTMLpreApplet = ''
var xHTMLpostApplet = ''
```

While still in the appbase.htm, search for *record* under the line Forms Applet Parameters, and add the *play* parameter. There will be three separate sections to add this *play* parameter. Changes need to be made to the *MACHtml*, *IEhtml* and the *NEhtml* parameters. While adding *play*, use the same format as displayed by the *record* parameter. This will allow all client browsers to accept the play feature. Below is an example of the additions made to the three parameters. Included are before and after examples. After these additions, save the file and now the play feature will be available.

### Example 1

Before:

```
// Forms Applet Parameters
MACHtml += '<' + 'PARAM name="serverPort" value="" + xsport + ">';
MACHtml += '<' + 'PARAM name="serverHost" value="" + xsname + xdname + ">';
;
MACHtml += '<' + 'PARAM name="connectMode" value="" + xconnectMode + ">';
MACHtml += '<' + 'PARAM name="serverApp" value="" + xserverApp + ">';
MACHtml += '<' + 'PARAM name="registryPath" value="" + xregistryPath + ">';
MACHtml += '<' + 'PARAM name="serverArgs" value="module=' + xmodule + ' use
rid=' + xuserid + ' fndnam=' + xfndnam ;
if (xrecord != "") {
    MACHtml += ' record=' + xrecord ;
}
if (xlog != "") {
    MACHtml += ' log=' + xlog;
}
```

After:

```
// Forms Applet Parameters
MACHtml += '<' + 'PARAM name="serverPort" value="" + xsport + ">';
MACHtml += '<' + 'PARAM name="serverHost" value="" + xsname + xdname + ">';
```

```

;
MACHtml += '<' + 'PARAM name="connectMode" value="' + xconnectMode + ">';
MACHtml += '<' + 'PARAM name="serverApp" value="' + xserverApp + ">';
MACHtml += '<' + 'PARAM name="registryPath" value="' + xregistryPath + ">';
MACHtml += '<' + 'PARAM name="serverArgs" value="module=' + xmodule + ' use
rid=' + xuserid + ' fndnam=' + xfndnam ;
if (xrecord != "") {
    MACHtml += ' record=' + xrecord ;
}
if (xplay != "") {
    MACHtml += ' play=' + xplay ;
}
if (xlog != "") {
    MACHtml += ' log=' + xlog;
}

```

### **Example 2**

*Before:*

// Forms Applet Parameters

```

IEhtml += '<' + 'PARAM name=serverPort value="' + xsport + ">';
IEhtml += '<' + 'PARAM name=serverHost value="' + xsname + xdname + ">';
IEhtml += '<' + 'PARAM name=connectMode value="' + xconnectMode + ">';
IEhtml += '<' + 'PARAM name=serverApp value="' + xserverApp + ">';
IEhtml += '<' + 'PARAM name=registryPath value="' + xregistryPath + ">';
IEhtml += '<' + 'PARAM name=serverArgs value="module=' + xmodule + ' userid='
+ xuserid + ' fndnam=' + xfndnam ;
if (xrecord != "") {
    IEhtml += ' record=' + xrecord ;
}
if (xlog != "") {
    IEhtml += ' log=' + xlog;
}

```

*After:*

// Forms Applet Parameters

```

IEhtml += '<' + 'PARAM name=serverPort value="' + xsport + ">';
IEhtml += '<' + 'PARAM name=serverHost value="' + xsname + xdname + ">';
IEhtml += '<' + 'PARAM name=connectMode value="' + xconnectMode + ">';
IEhtml += '<' + 'PARAM name=serverApp value="' + xserverApp + ">';
IEhtml += '<' + 'PARAM name=registryPath value="' + xregistryPath + ">';
IEhtml += '<' + 'PARAM name=serverArgs value="module=' + xmodule + ' userid='
+ xuserid + ' fndnam=' + xfndnam ;
if (xrecord != "") {
    IEhtml += ' record=' + xrecord ;
}
if (xplay != "") {
    IEhtml += ' play=' + xplay ;
}
if (xlog != "") {
    IEhtml += ' log=' + xlog;
}

```

### **Example 3**

*Before:*

```
// Forms Applet Parameters
NShtml += ' serverHost="' + xsname + xdname + ""';
NShtml += ' serverPort="' + xsport + ""';
NShtml += ' connectMode="' + xconnectMode + ""';
NShtml += ' serverApp="' + xserverApp + ""';
NShtml += ' registryPath="' + xregistryPath + ""';
NShtml += ' serverArgs="module='+ xmodule + ' userid=' + xuserid + ' fndnam='
+ xfndnam ;
if (xrecord != "") {
    NShtml += ' record=' + xrecord ;
}
if (xlog != "") {
    NShtml += ' log=' + xlog;
}
```

*After:*

```
// Forms Applet Parameters
NShtml += ' serverHost="' + xsname + xdname + ""';
NShtml += ' serverPort="' + xsport + ""';
NShtml += ' connectMode="' + xconnectMode + ""';
NShtml += ' serverApp="' + xserverApp + ""';
NShtml += ' registryPath="' + xregistryPath + ""';
NShtml += ' serverArgs="module='+ xmodule + ' userid=' + xuserid + ' fndn
+ xfndnam ;
if (xrecord != "") {
    NShtml += ' record=' + xrecord ;
}
if (xplay != "") {
    NShtml += ' play=' + xplay ;
}
if (xlog != "") {
    NShtml += ' log=' + xlog;
}
```

### **Recording a keystroke**

Steps to run the Record/Playback feature: First, record the session. To record this session include the record parameter and file name in the URL. Use the following example:

<http://host.domainname:port#/dev60cgi/f60cgi?record=/u09/oracle/play.tfi>.

Next, continue with the setups, test scripts, that you wish to record. Save and log out. The recorded session will continue until you log out. Every time you record a session, the file will be overwritten. If you wish to have numerous recorded files, change the name of the file in the URL with a description that is easy to understand. This will allow easy recognition of the file when it needs to be referenced. *(Note: It is best to use the keystrokes instead of the mouse. If you are not familiar with the different keystrokes, follow the path Help>Keyboard Help in the applications and you will see a list of keystrokes and their functions. Sometimes the keystroke option will not be available, use the mouse but we seen the playback die because the mouse was used in excess.)*

This is one method to record keystrokes. There are other methods to record keystrokes in a session depending on a desired level. One can record a session by leaving the record parameter blank in the appsweb.cfg file. Leaving this parameter blank will allow any user to record a session each and every time any user logs in to the applications. An advantage with using this level is the convenience, as all users can use their normal login pattern, which will enable the record feature to start. The disadvantage: Each time the user logs in; the previous file will be overwritten unless it has been renamed. Another way to record keystrokes is to change the profile setting of the ICX:FORMS\_LAUNCHER at the user site level. With this method, one can setup a user designated just for the record and playback feature. Define the designated user's profile setting, ICX:FORMS\_LAUNCHER, at the user site level, to include the URL with the record path as described in the previous example. The recorded session will start each time the designated user logs in through the home page. Once again, be careful as the file is overwritten each time the designated user logs into the applications.

### **Edit the forms event file**

After the session is completed, a forms event file will be created in the location designated by the URL. If no path is assigned, the default location will be where the forms server startup script is located. Edit the forms event file, if needed. Comments can be added to this file. Add a comment as a reminder to what changes you have just made. Also, comments can be added to this file for documentation purposes. This will allow order when an organization needs to make numerous file recordings to be played back. It will add greater ease in creating a plan to decide the hierarchy of the recorded files. *Security Alert: the user name and password are broadcasted in this file. Be careful who has permissions to this file because the password is not encrypted.*

### **Playback the recorded file**

After the previous steps are completed you are now in position to play back keystrokes. Include play, and the name of the file in the URL as follows:

<http://host.domainname:port#/dev60cgi/f60cgi?play=/u09/oracle/play.tfi>.

The automated playback will continue to work its way through each step until all the steps in the requested file are completed. Check it out and see if the playback was successful. Test it and run a report. Check and see if a new report is created. You can record the keystrokes as they create a new user. They save the new user and log out. Go to the newly recorded file and change the username. Now play it back. A new user has just been created. Remember, as you enter the responsibility for the user it may be tempting to use the mouse, but use tab key to pass through the blank fields and then enter the responsibility. It operates much cleaner when using the keys versus mouse.

### **Scenarios**

This Record/Playback feature can be used in many areas, including: setups, testing, training instance refreshing, and cloning. Below are a couple of scenarios that would be practical for use of the Record/Playback feature.

#### **Application Setups**

While doing setups, an error has occurred. When support is called, a patch is recommended. Now, the DBA applies the patch. This can be passed back and forth between a Functional team member, and the DBA several times before the issue is resolved. A better solution would be to record the setup. Let the DBA troubleshoot the issue and play back the setups until the issue is resolved. This could save a lot of time for both the DBA and Functional staff.

### **Application Testing**

It also could be valuable in testing. Due to the automation it performs the replay of the setups much faster and avoids manual entry mistakes. To validate the testing, one may have to change an entry in the form event file. In performing the testing one needs to add an invoice. However, each invoice needs to be unique. Simply, edit the recorded forms event file and change the invoice # allowing the playback to run without error.

### **Application Training Instance Refresh**

During preparation for in-house training its typical to populate the applications with seeded training data. Instead of reentering training information each time, record sessions for playback after each class. As a class fills up the training instance, and its refreshed each week, or at the end of each training class, the database for the training instance is refreshed with a copy of production, or possibly a “gold training” instance. Once the refreshed database is restored, run the scripts to repopulate the seed data. This will save hours of time and effort.

### **Review**

The Record/Playback feature is a welcome addition to both technical and functional personnel’s toolbox. It is a very efficient way to re-execute setups, testing scenarios, and refresh your training instance. In a matter of moments, the Record/Playback feature can be added to you Oracle Applications 11i environment. It is a tremendous time saver, and when its use is perfected, will become invaluable tool for many.