

# Oracle Fusion Middleware 12c

## Oracle ReportsとOracle Formsの統合

Oracle ホワイト・ペーパー | 2016年5月



## 目次

概要	1
Oracle Forms	1
Oracle Reports	2
Oracle Forms の RUN_REPORT_OBJECT 組込み	2
RUN_REPORT_OBJECT の使用方法	3
RUN_REPORT_OBJECT の例	3
RUN_REPORT_OBJECT によるパラメータ・リストの使用	7
パラメータ・フォームが表示されるレポートの呼出し	8
“PFACTION”による問題の解決	8
PL/SQL ファンクションによる URL パラメータのエンコード	9
パラメータ・フォームを使った Reports の呼出し手順の構築	10
汎用的なプロシージャの呼出し方法の例	13
Oracle Forms の WEB.SHOW_DOCUMENT 組込み	14
WEB.SHOW_DOCUMENT の構文	15
WEB.SHOW_DOCUMENT を使ったレポートの呼出し	15
ユーザー名とパスワードの非表示	16
結論	17

## 概要

このホワイト・ペーパーでは、Oracle Reports と Oracle Forms の統合方法について説明します。このホワイト・ペーパーの目的は次のとおりです。

- » Oracle Forms の RUN\_REPORT\_OBJECT 組込みの使用方法を理解する。
- » Oracle Forms アプリケーションからのレポートのリクエスト方法を理解する。
- » Oracle Forms から完成したレポートを取得する方法を理解する。
- » Reports パラメータ・フォームが含まれるレポートを呼び出す方法を理解する。
- » シングル・サインオン (SSO) が有効な場合に、Oracle Forms から Oracle Reports を呼び出す方法を理解する。

このドキュメントは、Oracle Forms アプリケーション・コードの記述方法に関する実践的な知識を持ち、中間層での Oracle Forms と Oracle Reports の基本機能を理解している方を対象としています。また、Oracle HTTP Server と Oracle WebLogic Server の機能のある程度理解していることが望ましいでしょう。このドキュメントに含まれるサンプル・コードは説明と教育のみを目的としています。このコードを本番アプリケーションで使用する場合は、必ず使用前に徹底的かつ完全なテストを行ってください。

## Oracle Forms

Oracle Forms は、2 つのハイ・レベルなコンポーネントで構成されています。Oracle Forms Developer の設計時コンポーネント (別名 Form Builder) と、Forms Services デプロイメント (別名 Forms ランタイム) コンポーネントである Oracle Fusion Middleware です。このドキュメントの目的に基づき、Oracle Forms アプリケーションからのレポートの呼出しに必要な機能と組込みについてのみ説明します。

Oracle Forms からの Oracle Reports の呼出しに対応している Oracle Forms 組込みは次の 2 つです。

- » RUN\_REPORT\_OBJECT
- » WEB.SHOW\_DOCUMENT

これらの組込みについては、Oracle Forms Builder のオンライン・ヘルプで詳しく説明しています。これらの組込みを使用して Oracle Reports を呼び出す方法については、後で説明します。Oracle Forms のデプロイについて詳しくは、『Oracle Forms Deployment Guide』を参照してください。このガイドは、[Oracle Technology Network](#) (OTN) の Fusion Middleware 12c のドキュメント・ライブラリに含まれています。

以前のバージョンの Oracle Forms (特にバージョン 6.x 以前) からアプリケーションを移行している最中の場合、Oracle Forms と Oracle Reports の統合に組込みの RUN\_PRODUCT が使用されている場合、さらに RUN\_REPORT\_OBJECT を使用してコードをリライトできない (またはリライトしたくない) 場合は、Forms Migration Assistant (FMA) の使用方法に関するドキュメントを参照してください。この情報は、Fusion Middleware ドキュメント・ライブラリのドキュメント・タ

イトル『Oracle Forms Upgrading Oracle Forms 6i to Oracle Forms 12c』に含まれています。

## Oracle Reports

Oracle Forms と同様に、Oracle Reports も基本的に設計時ツール (Oracle Reports Builder および Oracle Fusion Middleware) で構成されています。Oracle Fusion Middleware は、デプロイメント用の Reports Server コンポーネントです。Fusion Middleware 内のデプロイメント・コンポーネントは、Oracle Reports Services または Oracle Reports Server と呼ばれます。このホワイト・ペーパーでは、Reports Services と Reports Server という用語を、同じコンポーネントで交互に使用します。

Oracle Reports のデプロイについて詳しくは、Oracle Reports のデプロイメント・ガイド『Publishing Reports with Oracle Reports Services』を参照してください。このガイドは、OTN の Fusion Middleware 12c ドキュメント・ライブラリに含まれます。

## Oracle FormsのRUN\_REPORT\_OBJECT組込み

Oracle Forms から Oracle Reports を呼び出すもっともセキュアな方法は、RUN\_REPORT\_OBJECT 組込みを使用することです。ユーザーのデータベース接続は、中間層サーバーで Oracle Forms から Oracle Reports に明示的に渡されるため、このような情報が URL で渡される場合のように傍受されるリスクはありません。

---

Oracle Forms で Oracle Reports を呼び出すには、Forms の環境設定ファイル (default.env など) で、新しい環境変数を設定する必要があります。COMPONENT\_CONFIG\_PATH を、Reports ツール・コンポーネントの完全修飾パスに設定します。たとえば、次のとおりです。

```
COMPONENT_CONFIG_PATH=DOMAINHOME/config/fmwconfig/components/ReportsToolsComponent/<reports_tools_component_name>
```

---

Oracle Forms Builder で RUN\_REPORT\_OBJECT 組込みを使用するには、オブジェクト・ナビゲータの“Reports”ノードに新しい Reports オブジェクトを作成する必要があります。Reports オブジェクトごとに論理名があり、Forms 内で PL/SQL からレポートを呼び出すために使用されます。Reports の物理ファイルごとに、オプションで新しい Reports オブジェクトを作成できます。1 個の Reports オブジェクトを、Reports の多くの物理ファイルで使用することもできます。このオブジェクトの属性は、設計時に Builder のプロパティ・パレットで設定することも、実行時にプログラマ的に設定することもできます。

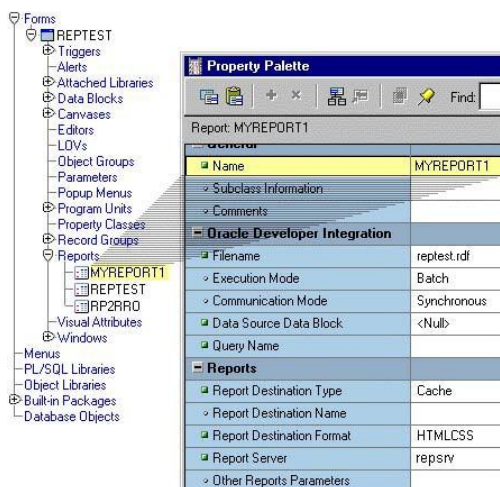


図1：Oracle Formsのオブジェクト・ナビゲータとプロパティ・パレット。“Reports”ノードには、オブジェクト“MYREPORT1”、“REPTTEST”、“RP2RRO”が含まれています。“MYREPORT1”オブジェクトによって参照されるOracle Reportsの物理ファイルは、“reptest.rdf”と定義されています。プロパティ・パレットの“Reports”ノードの下のOracle Reportsの実行時の設定は、実行時にSET\_REPORT\_OBJECT\_PROPERTYを使って上書きできます。

## RUN\_REPORT\_OBJECTの使用法

RUN\_REPORT\_OBJECT を使ってリモートの Reports Server にアクセスするには、Oracle Forms の Report オブジェクトが、Oracle Reports Services にアクセスできる必要があります。この操作は、SET\_REPORT\_OBJECT\_PROPERTY 組込みを使って動的に行うことも、Report オブジェクトのプロパティ・パレットに Oracle Reports Server 名の文字列を入力して静的に行うこともできます。

Oracle Forms Services と Oracle Reports Services が正しく機能するには、同じネットワーク・サブネット内に存在する必要があるという点にも注意してください。これらが同じネットワーク・サブネットに存在しない場合は、Oracle Reports ネーミング・サービスまたは Oracle Reports ブリッジを使用して、この特定の構成の制限事項を解決することができます。Reports のネーミング・サービスやブリッジの使用について詳しくは、前述のドキュメント『Publishing Reports with Oracle Reports Services』を参照してください。

## RUN\_REPORT\_OBJECTの例

### 例 1

次の例では、Oracle Forms の組込みである RUN\_REPORT\_OBJECT を使ってレポートを実行しています。この時点では、レポートの実行のみをリクエストしています。取得したデータ（レポート出力）は、この時点ではエンドユーザーに返されません。この仕様のほうが望ましい場合があります。その場合は DESTYPE を“FILE”に設定して、ファイルを後で使用できるように永続的にサーバーに保存します。

この例では、Reports オブジェクトの名前は“MyReport1”です。ユーザーが定義した Reports パラメータ“p\_deptno”が、“dept.deptno”フィールドの値を使って渡されます。このパラメータ・フォームは、“paramform=no”の使用より優先されます。

```
DECLARE
    report_id Report_Object;
    ReportServerJob
    VARCHAR2(254);
BEGIN
    report_id := find_report_object('MyReport1');
    SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_COMM_MODE,SYNCHRONOUS);
    SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_DESTTYPE,CACHE);
    SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_DESFORMAT,'PDF');
    SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_SERVER,'Repsrv');
    SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_OTHER,'p_deptno='||:Dept.Deptno||' paramform=no');
    ReportServerJob := run_report_object(report_id);
END;
```

図2 : RUN\_REPORT\_OBJECTの一般的な使用方法

## 例 2

次の例では、同期コールを使用して RUN\_REPORT\_OBJECT を呼び出して、レポートを実行します。ここでは、Reports オブジェクト名、Reports Server 名、および任意の出力形式（PDF、HTML、HTMLCSS など）がパラメータとして渡されます。また、レポートが正しく生成されたかどうかについて検証が試行され、結果がブラウザでエンドユーザーに表示されます。アプリケーションがアプリケーション内のさまざまな場所から Reports を呼び出すことが多い場合は、このような手順を使用することを推奨します。

```

PROCEDURE RUN_REPORT_OBJECT_PROC (vc_reportoj Varchar2, vc_reportserver varchar2, vc_runformat varchar2) IS
    v_report_id Report_Object;
    vc_ReportServerJob VARCHAR2(100); /* 各 Report リクエストの一意の ID */
    vc_rep_status VARCHAR2(100); /* Report ジョブのステータス */
    vjob_id VARCHAR2(100); /* 数字のみの文字列としての job_id */
BEGIN
    /* Report オブジェクトを取得 */
    v_report_id:= FIND_REPORT_OBJECT(vc_reportoj);

    /* Reports Server のレポート出力形式と名前、およびユーザー定義のパラメータを定義する。Forms から Reports に部門番号を渡す。paramform が"no"に設定されているため、パラメータ・フォームの表示は不要。 */

    SET_REPORT_OBJECT_PROPERTY(v_report_id,REPORT_DESFORMAT,vc_runformat);
    SET_REPORT_OBJECT_PROPERTY(v_report_id,REPORT_DESTYPE,CACHE);
    SET_REPORT_OBJECT_PROPERTY(v_report_id,REPORT_COMM_MODE,SYNCHRONOUS);
    SET_REPORT_OBJECT_PROPERTY(v_report_id,REPORT_SERVER,vc_reportserver);
    SET_REPORT_OBJECT_PROPERTY(v_report_id,REPORT_OTHER,'p_deptno=||:dept.deptno||'paramform=no');

    vc_ReportServerJob:=RUN_REPORT_OBJECT(v_report_id);
    vjob_id := substr(vc_ReportServerJob,instr(vc_ReportServerJob,'_',-1)+1);

    /* レポートのステータスを確認する。これは同期コール (REPORT_COMM_MODE) であったため、ステータス・チェックでは FINISHED またはエラーのみが返される。COMM_MODE が"asynchronous"に設定されている場合は、実行中のレポートのステータスを表示する前に、その定期的な変更が必要である。 */
    vc_rep_status := REPORT_OBJECT_STATUS(vc_ReportServerJob);
    IF vc_rep_status = 'FINISHED' THEN

    /* ブラウザに表示する Reports 出力を呼び出す。関連するアドレッシングの URL は、Reports Server が Forms サーバーと同じホストにあり、同じポート経由でアクセスできる場合にのみ有効となる。リモートの Reports 環境にアクセスするには、完全修飾 URL (http://hostname:port) を使用する必要がある。 */

        WEB.SHOW_DOCUMENT ('/reports/rwservlet/getjobid'|| vjob_id
            ||'?server=||vc_reportserver,'_blank);
    ELSE
        message ('Report failed with error message '||vc_rep_status);
    END IF;
END;

```

図3 : RUN\_REPORT\_OBJECTを使った、Oracle Reportsの呼出しの統合

Oracle Forms または Oracle Reports 6i からアップグレードして WEB.SHOW\_DOCUMENT を呼び出す場合は、RUN\_REPORT\_OBJECT 組込みによって取得される Reports job\_ID を変更して、Reports Server 名が含まれないようにする必要があります。

上記の手順を使用するには、“When-Button-Pressed”トリガーまたはその他の該当するトリガーで、次の情報を渡します。

RUN\_REPORT\_OBJECT\_PROC (<'REPORT\_OBJECT'>, <'REPORT\_SERVER\_NAME'>, <'FORMAT'>)

REPORT_OBJECT	Report の rdf ファイル名が含まれる、Forms の Report オブジェクト名
REPORT_SERVER_NAME	Reports Server の名前
FORMAT	html   html   css   pdf   xml   delimited   rtf のいずれかの形式

図4 : RUN\_REPORT\_OBJECT\_PROCの使用が必要なパラメータ

Reports に対する同期コールを行った場合、サーバー上でのレポートの処理中は待機する必要があります。

Reports を長時間実行する場合は、REPORT\_COMM\_MODE プロパティを asynchronous、REPORT\_EXECUTION\_MODE を batch に設定して、レポートが非同期に実行されるようにすることを推奨します。たとえば、次のとおりです。

```
SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_EXECUTION_MODE,BATCH);
SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_COMM_MODE,ASYNCHRONOUS);
```

RUN\_REPORT\_OBJECT を呼び出した後に、タイマーを作成して、When-Timer-Expired トリガーで現在の REPORT\_OBJECT\_STATUS に関する定期的なチェックを実行する必要があります。レポートが生成されると、“When-Timer-Expired”トリガーによって WEB.SHOW\_DOCUMENT 組込みが呼び出され、一意の job\_ID が付いた Reports 出力ファイルがクライアントのブラウザに表示されます。

When-Timer-Expired トリガーからレポート・ステータスをどのようにチェックできるかについて、次に例を示します。

```
(...)
/*:global.vc_ReportServerJob はグローバルであることが必要。これは、
レポートを開始するトリガー・コードと、レポートのステータスをチェックするトリガー・コード When-Timer-Expired の間で、
レポートの job_id に関する情報が共有されるためである。*/

vc_rep_status:= REPORT_OBJECT_STATUS(global.vc_ReportServerJob);
IF vc_rep_status='FINISHED' THEN
    vjob_id :=
        substr(global.vc_ReportServerJob,length(reportserver)+2,length(global.vc_ReportServerJob));
    WEB.SHOW_DOCUMENT
        ('/reports/rwservlet/getjobid||:vjob_id||?server=||vc_reportserver,'_blank');
    DELETE_TIMER (timer_id); -- レポートが完了。これ以上のチェックは不要。
ELSIF vc_rep_status not in ('RUNNING','OPENING_REPORT','ENQUEUED') THEN
    message (vc_rep_status||' Report output aborted');
    DELETE_TIMER (timer_id); -- レポートが失敗。これ以上のチェックは不要。
END IF;
```

図5：Reportsに対する非同期コールの実行と、When-Timer-Expiredからのステータスのチェック

## RUN\_REPORT\_OBJECTによるパラメータ・リストの使用

RUN\_PRODUCT<sup>1</sup>組込み（Oracle Forms で使用する場合はサポートされなくなりました）によって、Reports のシステム・パラメータとユーザー定義パラメータがパラメータ・リストに渡されます。RUN\_REPORT\_OBJECT でも同じパラメータ・リストを使用できます。ただし、SET\_REPORT\_OBJECT\_PROPERTY 組込みで設定する必要があるシステム・パラメータは使用できません。

REPORT_EXECUTION_MODE	BATCH または RUNTIME <sup>2</sup>
REPORT_COMM_MODE	SYNCHRONOUS   ASYNCHRONOUS
REPORT_DESTYPE	FILE   PRINTER   MAIL   CACHE <sup>3</sup>
REPORT_DESFORMAT	HTML   HTMLCSS   PDF   RTF   XML   DELIMITED   SPREADSHEET <sup>3</sup>
REPORT_FILENAME	レポートのファイル名
REPORT_DESNAME	レポートの宛先名
REPORT_SERVER	Report Server の名前

図6：RUN\_REPORT\_OBJECTで使われるシステム・パラメータのリスト

既存のパラメータ・リストにシステム・パラメータの定義がすでに含まれていると、エラーが発生する場合があります。このような問題が発生しないようにするには、パラメータ・リスト自体を変更します。このためには、DESNAME および DESTYPE のエントリを削除するか、

```
DELETE_PARAMETER (<parameter list>,'<name>');
```

をコードに追加してから、SET\_REPORT\_OBJECT\_PROPERTY を使用します。RUN\_REPORT\_OBJECT でパラメータ・リストを使用する構文は次のとおりです。

```
ReportServerJob := RUN_REPORT_OBJECT (report_id,paramlist_id);
```

DESTYPE がパラメータ・リストと SET\_REPORT\_OBJECT\_PROPERTIES の両方で定義されていると、モジュールのコンパイルは防止されずに、実行が防止される場合があります。

<sup>1</sup> RUN\_PRODUCTによるReports出力の生成は、Oracle Forms 9.0.4以降ではサポートされていません。RUN\_PRODUCTを使ったReportsへの統合コールが含まれるFormsモジュールはコンパイルされません。『Forms Upgrade Guide』でForms Migration Assistance (FMA) の使用方法を参照するか、前述のようにコードの更新を検討します。

<sup>2</sup> Report\_Execution\_Modelはクライアント/サーバー機能です。Oracle Formsでは現在使用されていません。ただし、値をBATCHまたはRUNTIMEに設定する必要があります。

<sup>3</sup> DESTYPEとDESFORMATのその他の値については、Oracle Reportsのデプロイメント・ガイド『Publishing Reports with Oracle Reports Services』を参照してください。

## パラメータ・フォームが表示されるレポートの呼出し

前の例で、レポートのパラメータ・フォームを RUN\_ REPORT\_OBJECT から呼び出した場合は、生成されるレポートの HTML パラメータ・フォームの<ACTION>属性が空のため、レポートのパラメータ・フォームは機能しません。RUN\_REPORT\_OBJECT コールは、サーバー上の Oracle Reports に直接送信されます。このため、HTML パラメータ・フォームを生成するときに、Oracle Reports から Web 環境にアクセスして、アクション属性の設定に必要な情報を取得することはできません。

<ACTION>属性は、「submit」ボタンを押したときの処理を定義する、HTML の標準的な<FORM>タグの一部です。Oracle Reports パラメータ・フォームの<ACTION>属性には、「submit」ボタンが押された後のリクエストの処理に必要な、非表示のランタイム・パラメータが含まれる必要があります。この条件を上書きするには、追加コードが必要です。

### “PFACTION”による問題の解決

“PFACTION”は Oracle Reports のコマンドライン・パラメータです。RUN\_REPORT\_OBJECT を使って Oracle Forms から Oracle Reports を呼び出す場合に、PFACTION を使ってレポートのパラメータ・フォームに非表示のランタイム・パラメータを追加できます。値“pfaction”のパラメータの構文は、次のようになります。

```
<request URL_to_rwservlet>?_hidden_<encoded_original_url_query_string>
```

“request URL\_to\_rwservlet”の部分には、プロトコル、ホスト名、ポート、Oracle Reports の Web コンテキスト・パスおよび Oracle Reports Servlet の名前が含まれます。たとえば、次のとおりです。

```
http://someDomain.com:8888/reports/rwservlet
```

“encoded\_original\_url\_query\_string”の部分は、SET\_REPORT\_OBJECT\_PROPERTY を使って Oracle Forms から Oracle Reports に渡され、URL でエンコードされた Oracle Reports システムおよびアプリケーションのすべてのパラメータの複製です。たとえば、次のような Reports のコマンドライン・パラメータがあります。

```
destype=cache desformat=htmlcss userid=scott/tiger@orcl
```

これは、次のように“pfaction”パラメータに値として追加されます。

```
destype=cache%20desformat=htmlcss%20userid=scott%2Ftiger%40orcl
```

RUN\_REPORT\_OBJECT を使ってパラメータ・フォームが含まれるレポートを呼び出すには、次の手順を実行します。

1. プロトコル、サーバーのホスト名、サーバー・ポートを入力します
2. Oracle Reports Servlet の仮想パスと名前を入力します
3. URL で“pfaction”コマンド・パラメータのパラメータ値をエンコードします

次の Forms 組込みを使って、Oracle Forms Services から Reports に“pfaction”コマンドを渡します。

```
SET_REPORT_OBJECT_PROPERTY (rep_id,REPORT_OTHER, 'pfaction ...');
```

REPORT\_OTHER 組込みを使って Oracle Reports にアプリケーション・パラメータを渡す場合は、このアプリケーション・パラメータが pfaction パラメータにも含まれる必要があります。

#### PL/SQLファンクションによるURLパラメータのエンコード

“pfaction”パラメータは ACTION パラメータとして Oracle Reports HTML パラメータ・フォームに追加されるため、エンコードされていない文字が含まれていないことを確認して、エラーが発生しないようにすることが重要です。URL に埋め込んだ場合に問題が発生しう原因の 1 つが、空欄（空白）です。このため、ほとんどの開発者は URL で空欄を“%20”にエンコードします。以下のように、PL/SQL ファンクション“ENCODE”によって、“;”、“/”、“?”、“:”、“@”、“+”、“\$”、“/”、および“ ”（セミコロン、スラッシュ、疑問符、コロン、アットマーク、プラス記号、ドル記号、カンマ、および空白）の文字がエンコードされます。

```
FUNCTION ENCODE (URL_PARAMS_IN Varchar2) RETURN VARCHAR2 IS
    v_url          VARCHAR2(2000) := URL_PARAMS_IN; -- Url 文字列
    v_url_temp v_a VARCHAR2(4000) := ''; -- 一時的な URL 文字列
    v_b c          VARCHAR2(10); -- 変換変数 VARCHAR2(10); -- 変換変数 CHAR;
    i              NUMBER(10);
BEGIN
    FOR i IN 1..LENGTH(v_url) LOOP
        c:= substr(v_url,i,1);
        IF c in (';', '/', '?', ':', '@', '+', '$', ',', ' ') THEN
            v_a := ltrim(to_char(trunc(ascii(substr(v_url,i,1))/16)));
            IF v_a = '10' THEN v_a := 'A';
                ELSIF v_a = '11' THEN v_a := 'B';
                ELSIF v_a = '12' THEN v_a := 'C';
                ELSIF v_a = '13' THEN v_a := 'D';
                ELSIF v_a = '14' THEN v_a := 'E';
                ELSIF v_a = '15' THEN v_a := 'F';
            END IF;
            v_b := ltrim(to_char(mod(ascii(substr(v_url,i,1)),16)));
            IF v_b = '10' THEN v_b := 'A';
                ELSIF v_b = '11' THEN v_b := 'B';
                ELSIF v_b = '12' THEN v_b := 'C';
                ELSIF v_b = '13' THEN v_b := 'D';
                ELSIF v_b = '14' THEN v_b := 'E';
                ELSIF v_b = '15' THEN v_b := 'F';
            END IF;
            v_url_temp := v_url_temp||'%'||v_a||v_b;
        ELSE
            v_url_temp :=v_url_temp||c;
        END IF;
    END LOOP;
    return v_url_temp;
END;
```

図7： PL/SQLファンクションによる文字列のURLエンコード

## パラメータ・フォームを使ったReportsの呼出し手順の構築

RUN\_REPORT\_OBJECT を使った Oracle Reports の呼出し処理には、Forms で汎用的な PL/SQL プロシージャを使用することが最適です。Oracle Forms からレポートを呼び出すには、Reports の pfaction コマンドに少なくとも次の情報を渡す必要があります。

- » Desformat : Reports の出力形式を決定します
- » Destype : 出力デバイス（プリンタまたはキャッシュ）を決定します
- » Userid : Reports とデータベースを接続します
- » Reports のファイル名 : 実行する Reports モジュールを指定します
- » Paramform = yes : Reports パラメータ・フォームがクライアント・ブラウザに表示されるようにします

RUN\_REPORT\_OBJECT を使って Oracle Reports に対するコールを処理する汎用的な PL/SQL プロシージャでは、前述したように、Reports オブジェクト・ノードが Forms で作成されている必要があります。1 つの Reports オブジェクト・ノードで、あらゆるレポートを実行できます。

このプロシージャでは、次の引数を使用されます。

report_id	FIND_REPORT_OBJECT('<name>');に対するコールで取得される Report オブジェクト
report_server_name	Reports Server の名前（例：“repserv11g”）
report_format	HTML、HTMLCSS、PDF、RTF、XML <sup>4</sup>
report_destype_name	CACHE、FILE、MAIL、PRINTER <sup>4</sup>
report_file_name	拡張子の付いた（または付かない）Reports ソース・モジュール
report_other_param	paramform やカスタム・アプリケーション・パラメータなどのその他のパラメータ。FILE または MAIL に出力する場合は、desname を‘otherparam’と入力する必要があります
report_servlet	Reports Servlet の仮想パス。相対パスとして指定しない仮想パスには、プロトコル（http または https）、ホスト名、ポート、Reports のコンテキスト・ルート（/reports）、およびサーブレットの名前（rwservlet）が含まれる必要があります。

図8：汎用的なプロシージャで渡されるパラメータの例

<sup>4</sup> DESTYPEとDESFORMATのその他の値については、Oracle Reportsのデプロイメント・ガイド『Publishing Reports with Oracle Reports Services』を参照してください。

```

PROCEDURE RUN_REPORT_OBJECT_PROC (
    report_id          REPORT_OBJECT,
    report_server_name VARCHAR2,
    report_format      VARCHAR2,
    report_destype_name NUMBER,
    report_file_name   VARCHAR2,
    report_otherparam  VARCHAR2,
    reports_servlet    VARCHAR2) ISBEGIN
report_message      VARCHAR2(100) := "";
rep_status          VARCHAR2(100) := "";
vjob_id             VARCHAR2(4000) := "";
hidden_action       VARCHAR2(2000) := "";
v_report_other      VARCHAR2(4000) := "";
i                   number (5);
c                   char;
c_old               char;
c_new               char;
BEGIN
-- Reports のランタイム・パラメータを設定する
SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_COMM_MODE,SYNCHRONOUS);
SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_FILENAME,report_file_name);
SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_SERVER,report_server_name);
SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_DESTYPE,report_destype_name);
SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_DESFORMAT,report_format);

-- pfaction パラメータの文字列を設定する
hidden_action := hidden_action || '&report=' ||
    GET_REPORT_OBJECT_PROPERTY(report_id,REPORT_FILENAME);
hidden_action := hidden_action || '&destype=' ||
    GET_REPORT_OBJECT_PROPERTY(report_id,REPORT_DESTYPE);
hidden_action := hidden_action || '&desformat=' ||
    GET_REPORT_OBJECT_PROPERTY (report_id,REPORT_DESFORMAT);

-- ユーザー名とパスワードの情報が、パラメータ・フォームの html ソース内に
-- 表示されることに注意。この情報は、シングル・サインオンを有効にすると表示されなくなる。
IF get_application_property (sso_userid) IS NULL Then
-- SSO ユーザーがない場合
hidden_action := hidden_action || '&userid=' ||
    get_application_property(username) || '/' ||
    get_application_property(password) || '@' ||
    get_application_property(connect_string);
ELSE
-- SSO ユーザーが識別された場合
hidden_action := hidden_action || '&ssoconn=' || get_application_property(config);
End if;
-- Reports の"other"パラメータは、キー値ペアとして渡される。例："key1=value1 key2=value2..."

```

図9：パラメータ・フォームを使用するレポートを呼び出す汎用的なプロシージャ (1/2)

-- 次のループによって、デリミタ付きの空白が Web で使用される"&"に置き換えられる。このように置き換えられるのは、  
-- 値自体に空白が含まれない場合のみ。これが要件の場合は、このコードを適切にカスタマイズする必要がある。

```
-- c_old はダミー値で初期化される
c_old := '@';
FOR i IN 1..LENGTH(report_otherparam) LOOP
    c_new := substr(report_otherparam,i,1);
    IF (c_new = ' ') THEN
        c := '&';
    ELSE
        c := c_new;
    END IF;
-- 複数の空白を削除する
    IF (c_old = ' ' and c_new = ' ') THEN
        null;
    ELSE
        v_report_other := v_report_other||c;
    END IF;
-- 現在の値を古い値として保存する
    c_old := c_new;
END LOOP;

hidden_action := hidden_action || '&' || v_report_other;

-- "reports_servlet"には Reports Servlet のパスが含まれる
-- 例 1 :
-- Forms と Reports が同じホストにあり、同じポート（8888 など）経由でアクセスされる
-- reports_servlet := '/reports/rwservlet'
-- 例 2 :
-- Forms と Reports が別のホストにあるか、別のポート（9001 と 9002 など）経由でアクセスされる
-- reports_servlet := 'http://host:port/reports/rwservlet'

hidden_action := reports_servlet||'?_hidden_server='||report_server_name|| encode(hidden_action);
SET_REPORT_OBJECT_PROPERTY (report_id,REPORT_OTHER,'pfaction='||
                            hidden_action||' '||report_otherparam);

-- Reports を呼び出す
report_message := run_report_object(report_id);
rep_status := report_object_status(report_message);
IF rep_status='FINISHED' THEN
    vjob_id :=substr(report_message,length(report_server_name)+2,length(report_message));
    WEB.SHOW_DOCUMENT(reports_servlet||'/getjobid'||vjob_id||'?server='||report_server_name,'_blank');
ELSE
-- ここでエラーを処理する
    message(rep_status);
END IF;

END;
```

図10：パラメータ・フォームを使用するレポートを呼び出す汎用的なプロシージャ（2/2）

### 汎用的なプロシージャの呼出し方法の例

Reports オブジェクト“reptest”が Forms モジュール内にある場合は、次のコードを WHEN-BUTTON-PRESSED トリガーで使用して、パラメータ・フォームが含まれるレポートを実行できます。

汎用的な PL/SQL プロシージャでは、Reports オブジェクトを引数として渡す必要があります。FIND\_REPORT\_OBJECT 組込みを使って、特定の Reports オブジェクト名の Reports オブジェクトを取得できます。

これは、Forms Services を実行するサーバーが Reports Services もホストすることが前提であるため、Reports Servlet パス変数の引数としてホスト名を渡す必要はありません。代わりに'/reports/rwervlet'を相対パスとして使用できます。Paramform=yes と設定すると、Reports で最初にパラメータ・フォームが表示されます。Reports の最終形式が PDF であっても、パラメータ・フォームは常に HTML で表示されます。

```
(...)  
-- Forms で Reports オブジェクトの ID を検索する  
report_id:=FIND_REPORT_OBJECT('reptest');  
-- 汎用的な PL/SQL プロシージャを呼び出して Reports を実行する  
-- この例で、同じホストとポートを使用して Forms と Reports の両方にアクセスする場合は、  
-- reports_server の値として相対パスのみを使用できることに注意する。  
RUN_REPORT_OBJECT_PROC( report_id,  
                        'repserve1-pc',  
                        'HTMLCSS',  
                        CACHE,  
                        'REPTTEST',  
                        'paramform=yes',  
                        '/reports/rwervlet');  
(...)
```

図10：パラメータ・フォームを使用するレポートを呼び出す汎用的なプロシージャ

以下の例では、追加のパラメータが Reports に渡されます。最初と 2 番目のキー値ペアの間のデリミタは、空白文字です。

```
(...)  
-- Forms で Reports オブジェクトの ID を検索する  
report_id:=FIND_REPORT_OBJECT('reptest');  
-- 汎用的な PL/SQL プロシージャを呼び出して Reports を実行する  
-- この例で、同じホストとポートを使用して Forms と Reports の両方にアクセスする場合は、  
-- reports_server の値として相対パスのみを使用できることに注意する。  
RUN_REPORT_OBJECT_PROC( report_id,  
                        'repserve1-pc',  
                        'HTMLCSS',  
                        CACHE,  
                        'REPTTEST',  
                        'paramform=no p_deptno=10',  
                        '/reports/rwervlet');  
(...)
```

図11：パラメータ・フォームが含まれないレポートの呼出しでも、p\_deptnoをランタイム・パラメータとして渡す必要がある。

Reports Services と Forms Services を別々のマシンやポート番号で実行する場合は、Reports Servlet の完全修飾 URL を入力する必要があります。次の例では、前述の PL/SQL プロシージャ RUN\_REPORT\_OBJECT\_PROC を使用しています。独自の PL/SQL コードを作成して、アプリケーションで“pfactions”コマンドを処理できます。

```
(...)  
-- Forms で Reports オブジェクトの ID を検索する  
report_id:=FIND_REPORT_OBJECT('reptest');  
-- 汎用的な PL/SQL プロシージャを呼び出して Reports を実行する  
-- この例で、同じホストとポートを使用して Forms と Reports の両方にアクセスする場合は、  
-- reports_server の値として相対パスのみを使用できることに注意する。  
RUN_REPORT_OBJECT_PROC(report_id,  
                        'repsserv1-pc',  
                        'HTMLCSS',  
                        CACHE,  
                        'REPTTEST',  
                        'paramform=yes',  
                        'http://someServer:9002/reports/rwsservlet');  
(...)
```

図12：パラメータ・フォームが含まれるが、Reports Servletが別のポート（またはホスト）をリスニングしているレポートの呼出し

## Oracle FormsのWEB.SHOW\_DOCUMENT組込み

WEB.SHOW\_DOCUMENT 組込みを使用して、Forms アプリケーションから Web サイト（URL）にアクセスします。この組込みによって、入力した URL が、クライアントのデフォルトの Web ブラウザに渡されます。Oracle Forms が宛先のブラウザを制御することはできません。組込みからブラウザに URL が渡されるだけです。この組込みを使って一部のローカル・コンテンツにアクセスすることもできますが、Web プロトコルとの組合せのみで使用することを推奨します。たとえば、この組込みと HTTP、HTTPS、FTP などのプロトコルの組合せが最適です。MAIL、MAILTO、FILE などの他のプロトコルでも使用できますが、推奨しません。このような他のプロトコルには、WebUtil などの他の方法でアクセスしてください。WebUtil については、Forms Builder のオンライン・ヘルプを参照してください。

## WEB.SHOW\_DOCUMENTの構文

WEB.SHOW\_DOCUMENT (URL, DESTINATION);

URL	URL は変数内の文字列 ('http://www.oracle.com')、または両方の組み合わせとして渡されます。アドレス指定した Web ページが Forms サーバーと同じホストにある場合は、相対パス ('/virtual_path/') を使用できます。
DESTINATION (別名 TARGET)	アドレス指定した Web ページが表示されるターゲットの定義です。値は一重引用符で囲みます。また小文字である必要があります。  _blank (デフォルト)  新規の名前の付いていないトップレベル・ウィンドウにドキュメントがロードされます。'windowName' と指定すると、windowName というウィンドウにドキュメントが表示されます。このウィンドウは、必要に応じて作成されます。  _self  ソースと同じフレームまたはウィンドウにドキュメントがロードされます。  _parent  ハイパーテキスト・リファレンスが含まれる親ウィンドウまたはフレームセットにドキュメントがロードされます。リファレンスがウィンドウまたはトップレベル・フレームにある場合は、target_self と同じです。  _top  ハイパーテキスト・リンクが含まれるウィンドウにドキュメントがロードされ、現在ウィンドウに表示されているフレームと置き換えられます。  <target name>  target_name で指定したフレームに Web ページが表示されます。

図13 : WEB.SHOW\_DOCUMENTの構文

## WEB.SHOW\_DOCUMENTを使ったレポートの呼出し

前の例では、RUN\_REPORT\_OBJECT を使ってレポートの生成をリクエストしてから、Reports の GETJOBID コマンドを WEB.SHOW\_DOCUMENT コールで呼び出して、そのレポートの出力を開く方法について説明しました。たとえば、次のとおりです。

```
WEB.SHOW_DOCUMENT(reports_servlet||'/getjobid'||vjob_id||'?server=||report_server_name,/_blank');
```

Oracle Forms から Oracle Reports を呼び出すもう 1 つの方法は、WEB.SHOW\_DOCUMENT を使って、フォームによって Reports Servlet に直接アクセスすることです。必要な URL では、次のような構文が使用されます。

```
http://<hostname>:<port>/reports/rwservlet?server=<reportserver>&report=<report>.rdf&desformat={html|css|pdf|xml|delimited}&destype=cache&userid=<user/pw@database>&paramform={no|yes}
```

次の例では、フォームからレポートを呼び出しています。ユーザー・パラメータ "p\_deptno" は、ブロック "dept." 内のフォームの項目 "deptno" から読み込まれることを前提とします。

```

/* WHEN-BUTTON-PRESSED */
DECLARE
    vc_url varchar2(200);
BEGIN
    vc_url:= 'http://<hostname><port>/reports/rwervlet?server=||
'Repsrv&report=reptest.rdf&desformat=htmlcss&destype=cache'||
'&userid=user/pw@database&p_deptno=||:dept.deptno||'&paramform=no';
    WEB.SHOW_DOCUMENT(vc_url,'_blank');
END;

```

図14：WEB.SHOW\_DOCUMENTによるReports Servletの呼出しの一般的な使用例

次の例は、Oracle Reports Server と Oracle Forms が同じホストにインストールされている場合の、相対アドレッシングの使用法を示したものです。

```

/* WHEN-BUTTON-PRESSED */
DECLARE
    vc_url varchar2(100);
BEGIN
    vc_url:= '/reports/rwervlet?server=Repsrv&report=reptest.rdf&desformat=htmlcss'||
'&destype=cache&userid=user/pw@database&p_deptno=||:dept.deptno||'&paramform=no';
    WEB.SHOW_DOCUMENT(vc_url,'_blank');
END;

```

図15：相対パスによるFormsからReports Servletへのアクセス

## ユーザー名とパスワードの非表示

この方法でレポートを実行するには、リクエスト URL の一部としてデータベース接続情報を渡す必要があります。ユーザーの機密情報を URL リクエストに追加することは、セキュリティ上重大な問題があります。ユーザーからリクエストされるすべての URL は、ブラウザの URL 履歴での検索やネットワーク・トラフィックでの取得が簡単であるためです。この問題を回避するには、シングル・サインオンを実装して、ユーザー名とパスワードをリクエストから除外します。

シングル・サインオン・ソリューションの実装方法について詳しくは、お使いのバージョンの Fusion Middleware のドキュメント・ライブラリを参照してください。

## 結論

Oracle Forms と Oracle Reports は、RUN\_REPORT\_OBJECT 組込み、または Forms の WEB.SHOW\_DOCUMENT 組込みを使用して統合できます。Oracle Reports Services にレポート・パラメータをセキュアに渡すには、RUN\_REPORT\_OBJECT 組込みを使用します。RUN\_REPORT\_OBJECT の使用は、出力をすぐにエンドユーザーに提示しないレポートを、シームレスにリクエストする場合にも適しています。WEB.SHOW\_DOCUMENT は、シンプルなリクエスト、および機密データやユーザー情報を渡す必要のないリクエストに適しています。

シングル・サインオンを使用すると、レポートを呼び出した Oracle Forms アプリケーションへのアクセス認証を受けているユーザーは、Oracle Reports Services の実行についても自動的に認証されます。

このトピックについて詳しくは、Forms Builder のオンライン・ヘルプと、Forms のデプロイメント・ガイドを参照してください。



#### Oracle Corporation, World Headquarters

500 Oracle Parkway  
Redwood Shores, CA 94065, USA

#### 海外からのお問い合わせ窓口

電話：+1.650.506.7000  
ファクシミリ：+1.650.506.7200

#### CONNECT WITH US

-  [blogs.oracle.com/oracle](https://blogs.oracle.com/oracle)
-  [facebook.com/oracle](https://facebook.com/oracle)
-  [twitter.com/oracle](https://twitter.com/oracle)
-  [oracle.com](https://oracle.com)

#### Integrated Cloud Applications & Platform Services

Copyright © 2016, Oracle and/or its affiliates. All rights reserved. 本文書は情報提供のみを目的として提供されており、記載内容は予告なく変更されることがあります。本文書は一切間違いがないことを保証するものではなく、さらに、口述による明示または法律による黙示を問わず、特定の目的に対する商品性もしくは適合性についての黙示的な保証を含み、いかなる他の保証や条件も提供するものではありません。オラクルは本文書に関するいかなる法的責任も明確に否認し、本文書によって直接的または間接的に確立される契約義務はないものとします。本文書はオラクルの書面による許可を前もって得ることなく、いかなる目的のためにも、電子または印刷を含むいかなる形式や手段によっても再作成または送信することはできません。

Oracle および Java は Oracle およびその子会社、関連会社の登録商標です。その他の名称はそれぞれの会社の商標です。

Intel および Intel Xeon は Intel Corporation の商標または登録商標です。すべての SPARC 商標はライセンスに基づいて使用される SPARC International, Inc. の商標または登録商標です。AMD、Opteron、AMD ロゴおよび AMD Opteron ロゴは、Advanced Micro Devices の商標または登録商標です。UNIX は、The Open Group の登録商標です。0116

ホワイト・ペーパー・タイトル 2016 年 5 月

著者：Oracle Product Management

共著者：Michael Ferrante、Frank Nimphius

共著者：Michael Ferrante、Frank Nimphius



Oracle is committed to developing practices and products that help protect the environment.