256
OWA_UTIL

The OWA_UTIL package contains utility subprograms for performing operations such as getting the value of CGI environment variables, printing the data that is returned to the client, and printing the results of a query in an HTML table.

> **See Also:**
>
> For more information about implementation of this package:
>
> - *Oracle Fusion Middleware Administrator's Guide for Oracle HTTP Server*
> - *Oracle Fusion Middleware User's Guide for mod_plsql*

This chapter contains the following topics:

- Overview
- Types
- Summary of OWA_UTIL Subprograms

OWA_UTIL Overview

The OWA_UTIL package contains three types of utility subprograms.

- Dynamic SQL Utilities enable you to produce pages with dynamically generated SQL code.
- HTML utilities enable you to retrieve the values of CGI environment variables and perform URL redirects.
- Date utilities enable correct date-handling. Date values are simple strings in HTML, but are treated as a datatype by the Oracle database.

OWA_UTIL Types

OWA_UTIL uses Types to specify creating information.

- DATETYPE Datatype
- IDENT_ARR Datatype
- IP_ADDRESS Datatype

DATETYPE Datatype

The TODATE Function converts an item of this type to the type DATE, which is understood and properly handled as data by the database. The procedure CHOOSE_DATE Procedure enables the user to select the desired date.

```
TYPE dateType IS TABLE OF VARCHAR2(10) INDEX BY BINARY_INTEGER;
```

Related Topics

- TODATE Function
  This function converts the DATETYPE Datatype to the standard Oracle `DATE` type.
- CHOOSE_DATE Procedure
  This procedure generates three HTML form elements that allow the user to select the day, the month, and the year.

IDENT_ARR Datatype

This datatype is used for an array.

```
TYPE ident_arr IS TABLE OF VARCHAR2(30) INDEX BY BINARY_INTEGER;
```

IP_ADDRESS Datatype

This datatype is used by the GET_CLIENT_IP Function in the OWA_SEC package.

```
TYPE ip_address IS TABLE OF INTEGER INDEX BY BINARY_INTEGER;
```

Related Topics

- GET_CLIENT_IP Function
  This function returns the IP address of the client.
- OWA_SEC
  The `OWA_SEC` package provides an interface for custom authentication.

Summary of OWA_UTIL Subprograms

This table lists the `OWA_UTIL` subprograms and briefly describes them.

Table 256-1 ***OWA_UTIL Package Subprograms***

Subprogram	Description
BIND_VARIABLES Function	Prepares a SQL query and binds variables to it
CALENDARPRINT Procedures	Prints a calendar
CELLSPRINT Procedures	Prints the contents of a query in an HTML table
CHOOSE_DATE Procedure	Generates HTML form elements that allow the user to select a date
GET_CGI_ENV Function	Returns the value of the specified CGI environment variable
GET_OWA_SERVICE_PATH Function	Returns the full virtual path for the PL/SQL Gateway
GET_PROCEDURE Function	Returns the name of the procedure that is invoked by the PL/SQL Gateway
HTTP_HEADER_CLOSE Procedure	Closes the HTTP header
LISTPRINT Procedure	Generates a HTML form element that contains data from a query
MIME_HEADER Procedure	Generates the Content-type line in the HTTP header
PRINT_CGI_ENV Procedure	Generates a list of all CGI environment variables and their values
REDIRECT_URL Procedure	Generates the Location line in the HTTP header
SHOWPAGE Procedure	Prints a page generated by the HTP and HTF packages in SQL*Plus

Table 256-1 (Cont.) *OWA_UTIL Package Subprograms*

Subprogram	Description
SHOWSOURCE Procedure	Prints the source for the specified subprogram
SIGNATURE procedure	Prints a line that says that the page is generated by the PL/SQL Agent
STATUS_LINE Procedure	Generates the Status line in the HTTP header
TABLEPRINT Function	Prints the data from a table in the database as an HTML table
TODATE Function	Converts dateType data to the standard PL/SQL date type
WHO_CALLED_ME Procedure	Returns information on the caller of the procedure.

BIND_VARIABLES Function

This function prepares a SQL query by binding variables to it, and stores the output in an opened cursor. Use this function as a parameter to a procedure sending a dynamically generated query. Specify up to 25 bind variables.

Syntax

```
OWA_UTIL.BIND_VARIABLES(
   theQuery       IN       VARCHAR2   DEFAULT NULL,
   bv1Name        IN       VARCHAR2   DEFAULT NULL,
   bv1Value       IN       VARCHAR2   DEFAULT NULL,
   bv2Name        IN       VARCHAR2   DEFAULT NULL,
   bv2Value       IN       VARCHAR2   DEFAULT NULL,
   bv3Name        IN       VARCHAR2   DEFAULT NULL,
   bv3Value       IN       VARCHAR2   DEFAULT NULL,
        ...
   bv25Name       IN       VARCHAR2   DEFAULT NULL,
   bv25Value      IN       VARCHAR2   DEFAULT NULL)
 RETURN INTEGER;
```

Parameters

Table 256-2 BIND_VARIABLES Function Parameters

Parameter	Description
theQuery	The SQL query statement which must be a SELECT statement
bv1Name	The name of the variable
bv1Value	The value of the variable

Return Values

An integer identifying the opened cursor.

CALENDARPRINT Procedures

These procedures creates a calendar in HTML with a visible border. Each date in the calendar can contain any number of hypertext links.

This procedure has 2 versions.

- Version 1 uses a hard-coded query stored in a varchar2 string.
- Version 2 uses a dynamic query prepared with the BIND_VARIABLES Function.

Syntax

```
OWA_UTIL.CALENDARPRINT(
   p_query        IN       VARCHAR2,
   p_mf_only      IN       VARCHAR2    DEFAULT 'N');

OWA_UTIL.CALENDARPRINT(
   p_cursor       IN       INTEGER,
   p_mf_only      IN       VARCHAR2    DEFAULT 'N');
```

Parameters

Table 256-3   CALENDARPRINT Procedure Parameters

Parameter	Description
`p_query`	A PL/SQL query.
`p_cursor`	A PL/SQL cursor containing the same format as `p_query`.
`p_mf_only`	If "`N`" (the default), the generated calendar includes Sunday through Saturday. Otherwise, it includes Monday through Friday only.

Usage Notes

Design your query as follows:

- The first column is a DATE. This correlates the information produced by the query with the calendar output generated by the procedure.
- The query output must be sorted on this column using ORDER BY.
- The second column contains the text, if any, that you want printed for that date.
- The third column contains the destination for generated links. Each item in the second column becomes a hypertext link to the destination given in this column. If this column is omitted, the items in the second column are simple text, not links.

CELLSPRINT Procedures

This procedure generates an HTML table from the output of a SQL query. SQL atomic data items are mapped to HTML cells and SQL rows to HTML rows. You must write the code to begin and end the HTML table.

There are nine versions of this procedure:

- The first version passes the results of a query into an index table. Perform the query and CELLSPRINT does the formatting. To have more control in generating an HTML table from the output of an SQL query, use the FORMAT_CELL Function in the " HTF" package.
- The second and third versions display rows (up to the specified maximum) returned by the query or cursor.
- The fourth and fifth versions exclude a specified number of rows from the HTML table. Use the fourth and fifth versions to scroll through result sets by saving the last row seen in a hidden form element.
- The sixth through ninth versions are the same as the first four versions, except that they return a row count output parameter.

Syntax

```
OWA_UTIL.CELLSPRINT(
   p_colCnt           IN     INTEGER,
   p_resultTbl        IN     vc_arr,
```

p_format_numbers IN VARCHAR2 DEFAULT NULL);

```
OWA_UTIL.CELLSPRINT(
   p_theQuery         IN     VARCHAR2,
   p_max_rows         IN     NUMBER      DEFAULT 100,
   p_format_numbers   IN     VARCHAR2    DEFAULT NULL);

OWA_UTIL.CELLSPRINT(
   p_theCursor        IN     INTEGER,
   p_max_rows         IN     NUMBER      DEFAULT 100,
   p_format_numbers   iN     VARCHAR2    DEFAULT NULL);

OWA_UTIL.CELLSPRINT(
   p_theQuery         IN     VARCHAR2,
   p_max_rows         IN     NUMBER      DEFAULT 100,
   p_format_numbers   IN     VARCHAR2    DEFAULT NULL,
   p_skip_rec         IN     NUMBER       DEFAULT 0,
   p_more_data        OUT    BOOLEAN);

OWA_UTIL.CELLSPRINT(
   p_theCursor        IN     INTEGER,
   p_max_rows         IN     NUMBER      DEFAULT 100,
   p_format_numbers   IN     VARCHAR2    DEFAULT NULL,
   p_skip_rec         IN     NUMBER      DEFAULT 0,
   p_more_data        OUT    BOOLEAN);

OWA_UTIL.CELLSPRINT(
   p_theQuery         IN     VARCHAR2,
   p_max_rows         IN     NUMBER      DEFAULT 100,
   p_format_numbers   IN     VARCHAR2    DEFAULT NULL,
   p_reccnt           OUT    NUMBER);

OWA_UTIL.CELLSPRINT(
   p_theCursor        IN     INTEGER,
   p_max_rows         IN     NUMBER      DEFAULT 100,
   p_format_numbers   IN     VARCHAR2    DEFAULT NULL,
   p_reccnt           OUT    NUMBER);

OWA_UTIL.CELLSPRINT(
   p_theQuery         IN     VARCHAR2,
   p_max_rows         IN     NUMBER      DEFAULT 100,
   p_format_numbers   IN     VARCHAR2    DEFAULT NULL,
   p_skip_rec         IN     NUMBER      DEFAULT 0,
   p_more_data        OUT    BOOLEAN
   p_reccnt           OUT    NUMBER);

OWA_UTIL.CELLSPRINT(
   p_theCursor        IN     INTEGER,
   p_max_rows         IN     NUMBER      DEFAULT 100,
   p_format_numbers   IN     VARCHAR2    DEFAULT NULL,
   p_skip_rec         IN     NUMBER      DEFAULT 0,
   p_more_data        OUT    BOOLEAN,
   p_reccnt           OUT    NUMBER);
```

Parameters

Table 256-4 CELLSPRINT Procedure Parameters

Parameter	Description
`p_query`	A PL/SQL query.
`p_colCnt`	The number of columns in the table.
`p_theQuery`	A SQL `SELECT` statement.
`p_theCursor`	A cursor ID. This can be the return value from the BIND_VARIABLES Function.
`p_max_rows`	The maximum number of rows to print.
`p_format_numbers`	If the value of this parameter is not `NULL`, number fields are right justified and rounded to two decimal places.
`p_skip_rec`	The number of rows to exclude from the HTML table.
`p_more_data`	`TRUE` if there are more rows in the query or cursor, `FALSE` otherwise.
`p_reccnt`	The number of rows that have been returned by the query. This value does not include skipped rows (if any).
`p_resultTbl`	The index table which will contain the result of the query. Each entry in the query will correspond to one column value.

Examples

This procedure generates

```
<tr><td>QueryResultItem</td><td>QueryResultItem</td></tr>...
```

CHOOSE_DATE Procedure

This procedure generates three HTML form elements that allow the user to select the day, the month, and the year.

Syntax

```
OWA_UTIL.CHOOSE_DATE(
   p_name          IN       VARCHAR2,
   p_date          IN       DATE       DEFAULT SYSDATE);
```

Parameters

Table 256-5 CHOOSE_DATE Procedure Parameters

Parameter	Description
`p_name`	The name of the form elements.
`p_date`	The initial date that is selected when the HTML page is displayed.

Usage Notes

- The parameter in the procedure that receives the data from these elements must be a GET_CGI_ENV Function.

- Use the TODATE Function to convert the GET_CGI_ENV Function value to the standard Oracle DATE datatype.

Examples

```
<SELECT NAME="p_name" SIZE="1">
<OPTION value="01">1
     ...
<OPTION value="31">31
</SELECT>
-
<SELECT NAME="p_name" SIZE="1">
<OPTION value="01">JAN
     ...
<OPTION value="12">DEC
</SELECT>
-
<SELECT NAME="p_name" SIZE="1">
<OPTION value="1992">1992
     ...
<OPTION value="2002">2002
</SELECT>
```

GET_CGI_ENV Function

This function returns the value of the specified CGI environment variable.

Syntax

```
OWA_UTIL.GET_CGI_ENV(
   param_name       IN      VARCHAR2)
 RETURN VARCHAR2;
```

Parameters

Table 256-6   GET_CGI_ENV Function Parameters

Parameter	Description
`param_name`	The name of the CGI environment variable. It is case-insensitive.

Return Values

The value of the specified CGI environment variable. If the variable is not defined, the function returns `NULL`.

GET_OWA_SERVICE_PATH Function

This function returns the full virtual path of the PL/SQL Gateway that is handling the request.

Syntax

```
OWA_UTIL.GET_OWA_SERVICE_PATH
 RETURN VARCHAR2;
```

Return Values

A virtual path of the PL/SQL Gateway that is handling the request.

GET_PROCEDURE Function

This function returns the name of the procedure that is being invoked by the PL/SQL Gateway.

Syntax

```
OWA_UTIL.GET_PROCEDURE
 RETURN VARCHAR2;
```

Return Values

The name of a procedure, including the package name if the procedure is defined in a package.

HTTP_HEADER_CLOSE Procedure

This procedure generates a newline character to close the HTTP header.

Syntax

```
OWA_UTIL.HTTP_HEADER_CLOSE;
```

Return Values

A newline character, which closes the HTTP header.

Usage Notes

- Use this procedure if you have not closed the header by using the `bclose_header` parameter in calls such as MIME_HEADER Procedure, REDIRECT_URL Procedure, or STATUS_LINE Procedure
- The HTTP header must be closed before any `HTP.PRINT` or `HTP.PRN` calls.

LISTPRINT Procedure

This procedure generates an HTML selection list form element from the output of a SQL query.

There are two versions of this procedure.

- The first version contains a hard-coded SQL query.
- The second version uses a dynamic query prepared with the BIND_VARIABLES Function.

Syntax

```
OWA_UTIL.LISTPRINT(
   p_theQuery     IN       VARCHAR2,
   p_cname        IN       VARCHAR2,
   p_nsize        IN       NUMBER,
   p_multiple     IN       BOOLEAN   DEFAULT FALSE);

OWA_UTIL.LISTPRINT(
   p_theCursor    IN    INTEGER,
   p_cname        IN       VARCHAR2,
   p_nsize        IN       NUMBER,
   p_multiple     IN       BOOLEAN   DEFAULT FALSE);
```

Parameters

Table 256-7 LISTPRINT Procedure Parameters

Parameter	Description
`p_theQuery`	The SQL query.
`p_theCursor`	The cursor ID. This can be the return value from the BIND_VARIABLES Function.
`p_cname`	The name of the HTML form element.
`p_nsize`	The size of the form element (this controls how many items the user can see without scrolling).
`p_multiple`	Whether multiple selection is permitted.

Usage Notes

The columns in the output of the query are handled in the following manner:

- The first column specifies the values that are sent back. These values are for the `VALUE` attribute of the `OPTION` tag.
- The second column specifies the values that the user sees.
- The third column specifies whether or not the row is marked as `SELECTED` in the `OPTION` tag. If the value is not `NULL`, the row is selected.

Examples

```
<SELECT NAME="p_cname" SIZE="p_nsize">
<OPTION SELECTED value='value_from_the_first_column'>value_from_the_second_column
<OPTION SELECTED value='value_from_the_first_column'>value_from_the_second_column
     ...
</SELECT>
```

MIME_HEADER Procedure

This procedure changes the default MIME header that the script returns. This procedure must come before any `HTP.PRINT` or `HTP.PRN` calls to direct the script not to use the default MIME header.

Syntax

```
OWA_UTIL.MIME_HEADER(
   ccontent_type    IN       VARCHAR2   DEFAULT 'text/html',
   bclose_header    IN       BOOLEAN    DEFAULT TRUE,
   ccharset         IN       VARCHAR2   DEFAULT NULL);
```

Parameters

Table 256-8 MIME_HEADER Procedure Parameters

Parameter	Description
`ccontent_type`	The MIME type to generate

Table 256-8 (Cont.) MIME_HEADER Procedure Parameters

Parameter	Description
bclose_header	Whether or not to close the HTTP header. If `TRUE`, two newlines are sent, which closes the HTTP header. Otherwise, one newline is sent, and the HTTP header remains open.
ccharset	The character set to use.The character set only makes sense if the MIME type is of type 'text'. Therefore, the character set is only tagged on to the Content-Type header only if the MIME type passed in is of type 'text'. Any other MIME type, such as 'image', will not have any character set tagged on.

Examples

```
Content-type: <ccontent_type>; charset=<ccharset>
```

so that

```
owa_util.mime_header('text/plain', false, 'ISO-8859-4')
```

generates

```
Content-type: text/plain; charset=ISO-8859-4\n
```

PRINT_CGI_ENV Procedure

This procedure generates all the CGI environment variables and their values made available by the PL/SQL Gateway to the stored procedure.

Syntax

```
OWA_UTIL.PRINT_CGI_ENV;
```

Examples

This procedure generates a list in the following format:

```
cgi_env_var_name = value\n
```

REDIRECT_URL Procedure

This procedure specifies that the application server is to visit the specified URL. The URL may specify either a Web page to return or a program to execute.

Syntax

```
OWA_UTIL.REDIRECT_URL(
   curl            IN       VARCHAR2
   bclose_header   IN       BOOLEAN    DEFAULT TRUE);
```

Parameters

Table 256-9 REDIRECT_URL Procedure Parameters

Parameter	Description
curl	The URL to visit.
bclose_header	Whether or not to close the HTTP header. If TRUE, two newlines are sent, which closes the HTTP header. Otherwise, one newline is sent, and the HTTP header remains open.

Usage Notes

This procedure must come before any HTP procedure or HTF function call.

Examples

This procedure generates

```
Location: <curl>\n\n
```

SHOWPAGE Procedure

This procedure prints out the HTML output of a procedure in SQL*Plus.

The procedure must use the HTP or HTF packages to generate the HTML page, and this procedure must be issued after the HTP or HTF page-generating subprogram has been called and before any other HTP or HTF subprograms are directly or indirectly called.

Syntax

```
OWA_UTIL.SHOWPAGE;
```

Usage Notes

- This method is useful for generating pages filled with static data.
- This procedure uses the DBMS_OUTPUT package and is limited to 32767 characters for each line and an overall buffer size of 1,000,000 bytes.

Examples

The output of htp procedure is displayed in SQL*Plus, SQL*DBA, or Oracle Server Manager. For example:

```
SQL> set serveroutput on
SQL> spool gretzky.html
SQL> execute hockey.pass("Gretzky")
SQL> execute owa_util.showpage
SQL> exit
```

This would generate an HTML page that could be accessed from Web browsers.

SHOWSOURCE Procedure

This procedure prints the source of the specified procedure, function, or package. If a procedure or function which belongs to a package is specified, then the entire package is displayed.

Syntax

```
OWA_UTIL.SHOWSOURCE (
   cname      IN     VARCHAR2);
```

Parameters

Table 256-10   SHOWSOURCE Procedure Parameters

Parameter	Description
`cname`	The function or procedure whose source you want to show.

SIGNATURE procedure

This procedure generates an HTML line followed by a signature line on the HTML document.

If a parameter is specified, the procedure also generates a hypertext link to view the PL/SQL source for that procedure. The link calls the SHOWSOURCE Procedure.

Syntax

```
OWA_UTIL.SIGNATURE;

OWA_UTIL.SIGNATURE (
   cname        IN        VARCHAR2);
```

Parameters

Table 256-11   SIGNATURE Procedure Parameters

Parameter	Description
`cname`	The function or procedure whose source you want to show.

Examples

Without a parameter, the procedure generates a line that looks like the following:

```
This page was produced by the PL/SQL Agent on August 9, 2001 09:30.
```

With a parameter, the procedure generates a signature line in the HTML document that looks like the following:

```
This page was produced by the PL/SQL Agent on 8/09/01 09:30
View PL/SQL Source
```

STATUS_LINE Procedure

This procedure sends a standard HTTP status code to the client.

This procedure must come before any `htp.print` or `htp.prn` calls so that the status code is returned as part of the header, rather than as "content data".

Syntax

```
OWA_UTIL.STATUS_LINE(
   nstatus        IN       INTEGER,
   creason        IN       VARCHAR2   DEFAULT NULL,
   bclose_header  IN       BOOLEAN    DEFAULT TRUE);
```

Parameters

Table 256-12   STATUS_LINE Procedure Parameters

Parameter	Description
`nstatus`	The status code.
`creason`	The string for the status code.
`bclose_header`	Whether or not to close the HTTP header. If `TRUE`, two newlines are sent, which closes the HTTP header. Otherwise, one newline is sent, and the HTTP header remains open.

Examples

This procedure generates

```
Status: <nstatus> <creason>\n\n
```

TABLEPRINT Function

This function generates either preformatted tables or HTML tables (depending on the capabilities of the user's browser) from database tables.

Syntax

```
OWA_UTIL.TABLEPRINT(
   ctable         IN       VARCHAR2,
   cattributes    IN       VARCHAR2   DEFAULT NULL,
   ntable_type    IN       INTEGER    DEFAULT HTML_TABLE,
   ccolumns       IN       VARCHAR2   DEFAULT '*',
   cclauses       IN    VARCHAR2   DEFAULT NULL,
   ccol_aliases   IN       VARCHAR2   DEFAULT NULL,
   nrow_min       IN       NUMBER     DEFAULT 0,
   nrow_max       IN       NUMBER     DEFAULT NULL)
 RETURN BOOLEAN;
```

Parameters

Table 256-13 TABLEPRINT Function Parameters

Parameter	Description
`ctable`	The database table.
`cattributes`	Other attributes to be included as-is in the tag.
`ntable_type`	How to generate the table. Specify `HTML_TABLE` to generate the table using `<TABLE>` tags or `PRE_TABLE` to generate the table using the `<PRE>` tags. These are constants: • `HTML_TABLE CONSTANT INTEGER` := 1; • `PRE_TABLE CONSTANT INTEGER` := 2;
`ccolumns`	A comma-delimited list of columns from `ctable` to include in the generated table.
`cclauses`	`WHERE` or `ORDER BY` clauses, which specify which rows to retrieve from the database table, and how to order them.
`ccol_aliases`	A comma-delimited list of headings for the generated table.
`nrow_min`	The first row, of those retrieved, to display.
`nrow_max`	The last row, of those retrieved, to display.

Return Values

Returns `TRUE` if there are more rows beyond the `nrow_max` requested, `FALSE` otherwise.

Usage Notes

- `RAW` columns are supported, but `LONG RAW` columns are not. References to LONG RAW columns will print the result 'Not Printable'.
- Note that in this function, `cattributes` is the second rather than the last parameter.

Examples

For browsers that do not support HTML tables, create the following procedure:

```
CREATE OR REPLACE PROCEDURE showemps IS
  ignore_more BOOLEAN;
BEGIN
  ignore_more := OWA_UTIL.TABLEPRINT('emp', 'BORDER', OWA_UTIL.PRE_TABLE);
END;
```

Requesting a URL such as

```
http://myhost:7777/pls/hr/showemps
```

returns to the following to the client:

```
<PRE>
-----------------------------------------------------------------
| EMPNO |ENAME |JOB      |MGR    |HIREDATE    | SAL | COMM | DEPTNO |
-----------------------------------------------------------------
| 7369| SMITH  | CLERK   | 7902 | 17-DEC-80 | 800 |      | 20 |
| 7499| ALLEN  | SALESMAN| 7698 | 20-FEB-81 | 1600 | 300 | 30 |
| 7521| WARD   | SALESMAN| 7698 | 22-FEB-81 | 1250 | 500 | 30 |
| 7566| JONES  | MANAGER | 7839 | 02-APR-81 | 2975 |     | 20 |
| 7654| MARTIN | SALESMAN| 7698 | 28-SEP-81 | 1250 | 1400| 30 |
| 7698| BLAKE  | MANAGER | 7839 | 01-MAY-81 | 2850 |     | 30 |
| 7782| CLARK  | MANAGER | 7839 | 09-JUN-81 | 2450 |     | 10 |
| 7788| SCOTT  | ANALYST | 7566 | 09-DEC-82 | 3000 |     | 20 |
| 7839| KING   | PRESIDENT |    | 17-NOV-81 | 5000 |     | 10 |
| 7844| TURNER | SALESMAN| 7698 | 08-SEP-81 | 1500 | 0   | 30 |
| 7876| ADAMS  | CLERK   | 7788 | 12-JAN-83 | 1100 |     | 20 |
| 7900| JAMES  | CLERK   | 7698 | 03-DEC-81 | 950  |     | 30 |
| 7902| FORD   | ANALYST | 7566 | 03-DEC-81 | 3000 |     | 20 |
| 7934| MILLER | CLERK   | 7782 | 23-JAN-82 | 1300 |     | 10 |
-------------------------------------------------------------------
</PRE>
```

To view the employees in department 10, and only their employee ids, names, and salaries, create the following procedure:

```
CREATE OR REPLACE PROCEDURE showemps_10 IS
 ignore_more BOOLEAN;
begin
 ignore_more := OWA_UTIL.TABLEPRINT
    ('EMP', 'BORDER', OWA_UTIL.PRE_TABLE,
     'empno, ename, sal', 'WHERE deptno=10 ORDER BY empno',
     'Employee Number, Name, Salary');
END;
```

A request for a URL like

```
http://myhost:7777/pls/hr/showemps_10
```

would return the following to the client:

```
<PRE>
-------------------------------------
| Employee Number |Name    | Salary |
-------------------------------------
| 7782            | CLARK  | 2450   |
| 7839            | KING   | 5000   |
| 7934            | MILLER | 1300   |
-------------------------------------
</PRE>
```

For browsers that support HTML tables, to view the department table in an HTML table, create the following procedure:

```
CREATE OR REPLACE PROCEDURE showdept IS
 ignore_more BOOLEAN;
BEGIN
 ignore_more := oWA_UTIL.TABLEPRINT('dept', 'BORDER');
END;
```

A request for a URL like

```
http://myhost:7777/pls/hr/showdept
```

would return the following to the client:

```
<TABLE BORDER>
<TR>
<TH>DEPTNO</TH>
<TH>DNAME</TH>
<TH>LOC</TH>
</TR>
<TR>
```

```
<TD ALIGN="LEFT">10</TD>
<TD ALIGN="LEFT">ACCOUNTING</TD>
<TD ALIGN="LEFT">NEW YORK</TD>
</TR>
<TR>
<TD ALIGN="LEFT">20</TD>
<TD ALIGN="LEFT">RESEARCH</TD>
<TD ALIGN="LEFT">DALLAS</TD>
</TR>
<TR>
<TD ALIGN="LEFT">30</TD>
<TD ALIGN="LEFT">SALES</TD>
<TD ALIGN="LEFT">CHICAGO</TD>
</TR>
<TR>
<TD ALIGN="LEFT">40</TD>
<TD ALIGN="LEFT">OPERATIONS</TD>
<TD ALIGN="LEFT">BOSTON</TD>
</TR>
</TABLE>
```

A Web browser would format this to look like the following table:

DEPTNO	DNAME	LOC
`10`	`ACCOUNTING`	`NEW YORK`
`20`	`RESEARCH`	`DALLAS`
`30`	`SALES`	`CHICAGO`

TODATE Function

This function converts the DATETYPE Datatype to the standard Oracle `DATE` type.

Syntax

```
OWA_UTIL.TODATE(
   p_dateArray      IN     dateType)
 RETURN DATE;
```

Parameters

Table 256-14   TODATE Function Parameters

Parameter	Description
`p_dateArray`	The value to convert.

Related Topics

- DATETYPE Datatype
  The TODATE Function converts an item of this type to the type `DATE`, which is understood and properly handled as data by the database. The procedure CHOOSE_DATE Procedure enables the user to select the desired date.

WHO_CALLED_ME Procedure

This procedure returns information (in the form of output parameters) about the PL/SQL code unit that invoked it.

Syntax

```
OWA_UTIL.WHO_CALLED_ME(
   owner          OUT      VARCHAR2,
   name           OUT      VARCHAR2,
   lineno         OUT      NUMBER,
   caller_t       OUT      VARCHAR2);
```

Parameters

Table 256-15   WHO_CALLED_ME Procedure Parameters

Parameter	Description
owner	The owner of the program unit.
name	The name of the program unit. This is the name of the package, if the calling program unit is wrapped in a package, or the name of the procedure or function if the calling program unit is a standalone procedure or function. If the calling program unit is part of an anonymous block, this is NULL.
lineno	The line number within the program unit where the call was made.
caller_t	The type of program unit that made the call. The possibilities are: package body, anonymous block, procedure, and function. Procedure and function are only for standalone procedures and functions.