Simple Example

```
PROCEDURE simple AS
BEGIN
   htp.htmlOpen;
   htp.headOpen;
   htp.title('Simple Example');
   htp.headClose;
   htp.bodyOpen;
   htp.line;
   htp.header(1,'Simple Example');
   htp.line;
   htp.paragraph;
   htp.print('Today''s date is: '||
             to_char(sysdate, 'DD/MM/YYYY'));
   htp.br;
   htp.print('Today''s day  is:  ' ||
             to_char(sysdate, 'DAY'));
   htp.paragraph;
   htp.print('Ordinary <em>tags</em> can be ' ||
             'used in the strings ' ||
             'that we send.');
   htp.line;
   htp.address('Raj Sunderraman');
   htp.bodyClose;
   htp.htmlClose;
END;
```

http://tinman.cs.gsu.edu:9001/matxxxx/plsql/simple

```
<HTML>
<HEAD>
<TITLE>Simple Example</TITLE>
</HEAD>
<BODY>
<HR>
<H1>Simple Example</H1>
<HR>
<P>
Today's date is:  04/02/1998
<BR>
Today's day  is:  WEDNESDAY
<P>
Ordinary <em>tags</em> can be used in the strings
that we send.
<HR>
<ADDRESS>Raj Sunderraman</ADDRESS>
<HR>
</BODY>
</HTML>
```

Passing Parameters

GET method:

```
CREATE OR REPLACE PROCEDURE pl1(str in varchar2,
                                num in number) AS
BEGIN
   htp.htmlOpen;
   htp.headOpen;
   htp.title('Parameters');
   htp.headClose;
   htp.bodyOpen;
   htp.line;
   htp.header(1,'Parameter Passing');
   htp.line;
   for i in 1..num loop
     htp.print(i || '. ' || str);
     htp.br;
   end loop;
   htp.line;
   owa_util.signature('pl1');
   htp.bodyClose;
   htp.htmlClose;
END;

http://tinman.cs.gsu.edu:9001/
   matxxxx/plsql/pl1?str=Hello&num=4
```

POST method: typically used with HTML forms; parameters are passed to standard input (PL/SQL cartridge automatically takes care of passing parameters as long as form element name is same as parameter name).

Printing HTML Tables

```
cursor c1 is
   select   *
   from     catalog;

htp.tableOpen(cattributes => 'border=2 width=60%' );
htp.tableRowOpen;
htp.tableHeader('Course Number');
htp.tableHeader('Course Title');
htp.tableRowClose;
for cat_rec in c1 loop
  htp.tableRowOpen;
  htp.tableData(cat_rec.cno);
  htp.tableData(cat_rec.ctitle);
  htp.tableRowClose;
end loop;
htp.tableClose;
```

Processing HTML Forms

Two procedures to process a HTML form:

- One procedure to create the dynamic HTML page containing the form and
- The second to process the user inputs obtained by the first procedure. The second procedure receives a number of input parameters from the first procedure. It is important to name these input parameters with the names chosen for the GUI elements in the first procedure.

Forms can have the following types of elements:

- Single line input text fields (`htp.formText`).
- Single line input password fields (`htp.formPassword`).
- Checkboxes (`htp.formCheckbox`).
- Radio buttons (`htp.formRadio`).
- Submit buttons (`htp.formSubmit`).
- Text areas (`htp.formTextarea`). Used to create a multi-line input field.
- Selects. (`htp.formSelectOpen`, `htp.formSelectOption`, `htp.formSelectClose`). Used to allow the user to choose one or more of a set of alternatives described by textual labels. These are usually rendered as a pulldown, pop up, or a fixed size list.

```
procedure get_access as
begin
   htp.htmlOpen;
   htp.headOpen;
   htp.title('Get password');
   htp.header(1, 'Grade Book Access Page');
   htp.headClose;
   htp.bodyOpen;
   htp.formOpen(owa_util.get_owa_service_path ||
                'user_access.start_session');
   htp.print('Today is ' ||
             to_char(sysdate, 'Day') || '  ' ||
             to_char(sysdate, 'Dd / Mon / YYYY HH:MI AM'));
   htp.nl;

   htp.tableOpen;
   htp.tableRowOpen;
   htp.tableData(htf.strong('USER ID: '));
   htp.tableData(htf.formText('usid', 20, 50));
   htp.tableRowClose;
   htp.tableRowOpen;
   htp.tableData(htf.strong('PASSWORD: '));
   htp.tableData(htf.formPassword('passwd', 20, 50));
   htp.tableRowClose;
   htp.tableClose;

   htp.formSubmit(NULL, ' Proceed ');
   htp.formReset;
   htp.formClose;
   htp.bodyClose;
   htp.htmlClose;
Exception
   when others then
      htp.print('error in user_access.get_access');
end get_access;
```

```
procedure start_session(usid in varchar2   DEFAULT NULL,
                        passwd in varchar2 DEFAULT NULL) as
   user_buffer     varchar(15);
   passwd_buffer   varchar(15);
   time_buffer     date;
   auth_buffer     varchar(15);
   random_num     varchar(10);
   access_buffer  varchar2(30);
   bad_passwd     exception;
begin
   select A.userid, A.password, A.lastAccess, A.authority
   into   user_buffer, passwd_buffer, time_buffer,
          auth_buffer
   from   users A
   where  upper(A.userid)=upper(start_session.usid);

   if start_session.passwd != passwd_buffer then
      raise bad_passwd;
   end if;
   random_num := to_char(sysdate, 'SSSSS');
   access_buffer := user_buffer || ',' || random_num;
   update users
   set u_access = access_buffer,
       lastAccess = sysdate
   where users.userid=user_buffer;
   commit;
   menus.teacher_menu(access_buffer);
Exception
   when bad_passwd then
      htp.print('Bad password ');
   when no_data_found then
      htp.print('Invalid user');
   when others then
      htp.print('Error in user_access.start_session');
end start_session;
```

```
create table users (
   userid     varchar2(15),
   password   varchar2(15),
   lastaccess date,
   u_access   varchar(30), -- coded string
   authority  varchar2(15) -- TEACHER/STUDENT
);
```

Student-Menu

- *Display Scores*: The scores for the course selected in the pull down list of courses can be displayed in this option.
- *Change Password*: The password can be changed in this option.
- *Logout*: The user can logout in this option. The only action the system takes in this case is to set the *u_access* column value for this user back to `null`.

```
procedure student_menu(u_access in varchar2 default NULL,
                       id in number) as
   access_check varchar2(30);
   no_access    exception;
   cursor c1 is
     select courses.term,cno,courses.lineno
     from   courses,enrolls
     where  enrolls.sid = id and
            courses.term = enrolls.term and
            courses.lineno = enrolls.lineno;
begin
   if (u_access = NULL) then
      raise no_access;
   end if;
   access_check := user_access.check_access(u_access);
   if  (access_check != 'Student') then
      raise no_access;
   end if;
   htp.htmlOpen;
   htp.headOpen;
   htp.htitle('Student Menu');
   htp.headClose;
   htp.bodyOpen;
   htp.formOpen(owa_util.get_owa_service_path ||
               'menus.process_student_option');
   htp.formSelectOpen('tcl','Term: ');
   for c1_rec in c1 loop
     htp.formSelectOption('(' ||c1_rec.term||','||
                          c1_rec.cno||','||
                          c1_rec.lineno || ')');
   end loop;
   htp.formSelectClose;
   htp.nl;
```

```
   htp.formHidden('u_access', u_access);
   htp.formHidden('id', id);
   htp.formSubmit('opt','Display Scores');
   htp.formSubmit('opt','Change Password');
   htp.formSubmit('opt','Logout');
   htp.formclose;
   htp.bodyClose;
   htp.htmlClose;
Exception
   when no_access then
      htp.print('You do not have an access to this page');
   when others then
      htp.print('Error in menus.student_menu');
end student_menu;
```

```
function check_access(
     u_access in varchar2 default NULL)
   return varchar2 as
 return_value varchar(30);
 auth_buffer  varchar(20);
  no_access    exception;
begin
  if u_access is null
     then raise no_access;
  end if;
  select authority into auth_buffer
  from   users U
  where  U.u_access = check_access.u_access;
  return_value := auth_buffer;
  return(return_value);
 Exception
   when no_data_found then return('0');
   when no_access then return ('0');
   when others then
    htp.print('Error in user_access.check_access');
end check_access;
```

```
procedure process_student_option(
    u_access in varchar2 default NULL,
    opt in varchar2 default null,
    tcl in varchar2 default null,
    id in number) as
begin
  if (opt = 'Display Scores') then
    display.display_own_scores(id,tcl,u_access);
  elsif (opt = 'Change Password') then
    user_access.change_password(u_access);
  elsif (opt = 'Logout') then
    user_access.logout(u_access);
  end if;
end;
```

```
procedure display_own_scores(
   id in number,
   tcl in varchar2,
   u_access in varchar2 default NULL) as

   counter integer := 0;
   access_check varchar2(30);
   no_access    exception;
   trm scores.term%type;
   lnum scores.lineno%type;
   p1 integer;
   p2 integer;
   cursor c1 is
      select components.compname,points,maxpoints
      from   scores,components
      where  sid = id and scores.term = trm and
             scores.lineno = lnum and
             components.compname = scores.compname and
             components.term = scores.term and
             components.lineno = scores.lineno;
begin
   if (u_access = NULL) then
      raise no_access;
   end if;
   access_check := user_access.check_access(u_access);
   if (access_check != 'Student') then
               raise no_access;
   end if;
```

```
htp.htmlOpen;
htp.headOpen;
htp.title('Display Scores');
htp.header(1,'Student Scores');
htp.headClose;
htp.nl;
htp.bodyOpen;
p1 := instr(tcl,',',1,2)+1;
p2 := instr(tcl,')',1,1)-1;
trm := substr(tcl,2,instr(tcl,',',1)-2);
lnum := to_number(substr(tcl,p1,p2-p1+1));
htp.tableOpen(cattributes => 'border=2 width=60%' );
htp.tableRowOpen;
htp.tableHeader('Component Name');
htp.tableHeader('Points');
htp.tableHeader('Max Points');
htp.tableRowClose;
for scr_rec in c1 loop
  htp.tableRowOpen;
  htp.tableData(scr_rec.compname,'left');
  htp.tableData(scr_rec.points,'right');
  htp.tableData(scr_rec.maxpoints,'right');
  htp.tableRowClose;
  counter := counter + 1;
end loop;
htp.tableClose;
htp.nl;
htp.print('Number of records: ');
htp.print(counter);
htp.bodyClose;
htp.htmlClose;
```

```
Exception
   when NO_DATA_FOUND then
      htp.print('No records found');
   when no_access then
      htp.print('You are not logged on');
   when others then
      htp.print('Something wrong in display_own_scores');
      htp.br;
      htp.print('SQLCODE = ' || SQLCODE);
      htp.br;
      htp.print('SQLERRM = ' || SQLERRM);
end display_own_scores;
```

```
procedure logout(u_access in varchar2 default NULL) as
   auth_buffer varchar(20);
   no_access   exception;
begin
   auth_buffer := check_access(u_access);
   if (auth_buffer < '1')
      then raise no_access;
   end if;
   update users
      set u_access = null
      where u_access = logout.u_access;
   commit;
   htp.print('Thank you for using Grade Book Program');
Exception
   when no_access then
      htp.print('You are not logged in');
   when others then
      htp.print('Error in users_access.logout');
end logout;
```

Multivalued Parameters

```
procedure add_scores(
    u_access in varchar2 default NULL,
    term_in in varchar2 default NULL,
    lnum in number default NULL,
    cnum in varchar2 default NULL) as

   access_check       varchar2(30);
   no_access       exception;

   cursor c1 is
      select term, lineno, compname
      from   components
      where  term = term_in and lineno = lnum;
begin
   if (u_access = NULL) then
      raise no_access;
   end if;

   access_check := user_access.check_access(u_access);
   if (access_check != 'Teacher') then
      raise no_access;
   end if;
```

```
   htp.htmlOpen;
   htp.headOpen;
   htp.title('Selecting Component');
   htp.header(1,'Selecting Component');
   htp.headClose;
   htp.bodyOpen;
   htp.formOpen(owa_util.get_owa_service_path ||
                'add_all.process_scores');
   htp.nl;
   htp.formSelectOpen('comp',
            'Choose the course component: ');
   for crec in c1 loop
     htp.formSelectOption(crec.compname);
   end loop;
   htp.formSelectClose;
   htp.formHidden('term_in', term_in);
   htp.formHidden('cnum', cnum);
   htp.formHidden('lnum', lnum);
   htp.formHidden('u_access', u_access);
   htp.formSubmit;
   htp.formClose;
   htp.bodyClose;
   htp.htmlClose;
Exception
   when no_access then
     htp.print('You are not logged on');
   when others then
     htp.print('Something is wrong in add_scores');
end add_scores;
```

```
procedure process_scores (
    u_access in varchar2 default NULL,
    term_in in varchar2 default NULL,
    lnum in integer default NULL,
    comp in varchar2 default NULL,
    cnum in varchar2 default NULL) as

   access_check    varchar2(30);
   no_access       exception;

   cursor c1 is
      select distinct B.lname, B.fname,
             A.sid, A.term, A.lineno
      from enrolls A, students B
      where (A.term = term_in) and
            (A.lineno = lnum) and
            (A.sid = B.sid);
begin

   if (u_access = NULL) then
      raise no_access;
   end if;

   access_check := user_access.check_access(u_access);
   if (access_check != 'Teacher') then
      raise no_access;
   end if;
```

```
htp.htmlOpen;
htp.headOpen;
htp.title('Entering Scores');
htp.header(1,'Entering Scores for ' || cnum ||
              ',' || lnum || ',' || term_in);
htp.headClose;
htp.bodyOpen;
htp.formOpen(owa_util.get_owa_service_path ||
     'insert_all.insert_scores');
htp.formHidden('u_access', u_access);
htp.formHidden('term_in', term_in);
htp.formHidden('lnum', lnum);
htp.formHidden('comp', comp);
htp.formHidden('score_arr', '000');
htp.formHidden('ids', '0000');

htp.print('Component: ');
htp.print(comp);
htp.nl;
htp.nl;
```

```
   htp.tableOpen;
   for crec in c1 loop
     htp.tableRowopen;
     htp.tableData(htf.strong(crec.lname||', '||
                              crec.fname));
     htp.tableData(htf.formText('score_arr', 3, 3));
     htp.formHidden('ids', crec.sid);
     htp.tableRowClose;
   end loop;
   htp.tableClose;
   htp.formSubmit(NULL,'Enter the scores');
   htp.formclose;
   htp.bodyClose;
   htp.htmlClose;
Exception
   when no_access then
     htp.print('You are not logged on');
   when others then
     htp.print('Something is wrong in process_scores');
end process_scores;
```

```
procedure insert_scores(u_access in varchar2 default NULL,
  term_in in varchar2 default NULL,
  lnum in number default NULL,
  comp in varchar2 default NULL,
  score_arr in owa_util.ident_arr,
  ids in owa_util.ident_arr) as

   access_check    varchar2(30);
   no_access       exception;
   counter         integer;

begin
   if (u_access = NULL) then
      raise no_access;
   end if;

   access_check := user_access.check_access(u_access);
   if (access_check != 'Teacher') then
      raise no_access;
   end if;

   htp.htmlOpen;
   htp.headOpen;
   htp.title('Inserting Scores');
   htp.headClose;
   htp.bodyOpen;
   htp.nl;
```

```
counter := 2;
loop
  insert into scores(sid,term,lineno,compname,points)
  values(ids(counter),term_in,lnum,comp,
         score_arr(counter));
  counter := counter+1;
end  loop;
commit;

EXCEPTION
  when no_access then
     htp.print('You are not logged on');
     htp.anchor2('http://tinman.cs.gsu.edu:9999/' ||
       'ows-bin/book/owa/user_access.get_access',
       'Back to sign in page');
     htp.bodyClose;
     htp.htmlClose;
  when no_data_found then
    if (counter = 2) then
      htp.print('No DATA');
      htp.nl;
      htp.anchor2('http://tinman.cs.gsu.edu:9999/' ||
       'ows-bin/book/owa/add_all.select_course?' ||
       'u_access='||u_access, 'Back to Previous Menu');
      htp.nl;
      htp.bodyClose;
      htp.htmlClose;
```

```
      else
        htp.print('Inserted scores successfully');
        htp.nl;
        htp.anchor2('http://tinman.cs.gsu.edu:9999/' ||
         'ows-bin/book/owa/add_all.select_course?' ||
         'u_access='||u_access, 'Back to Previous Menu');
        htp.nl;
        htp.bodyClose;
        htp.htmlClose;
      end if;
    when others then
      htp.print('Other Error');
      htp.br;
      htp.anchor2('http://tinman.cs.gsu.edu:9999/' ||
         'ows-bin/book/owa/add_all.select_course?' ||
                   'u_access='||u_access,
                   'Back to the Main Menu');
      htp.bodyClose;
      htp.htmlClose;
end insert_scores;
```

PL/SQL Web Toolkit

The PL/SQL Toolkit consists of the following packages:

- `HTP`. A hypertext procedure generates a line in an HTML document that contains the HTML tag that corresponds to its name. For instance, the `htp.line` procedure generates the `<HR>` tag. The HTP procedures are grouped into the following categories:
  - *Print Procedures*:

    `p, print, prn, prints, ps`
  - *Head Related Tags*:

    `title, base, isindex, linkRel, linkRev, meta`

– *General Body Tags*:

line, hr, nl, br, header,
anchor, anchor2,
mailto, img, img2, para,
paragraph,
address, comment, preOpen,
preClose,
blockquoteOpen,
blockquoteClose, base,
area, mapOpen, mapClose,
bgsound, div,
listingOpen, listingClose,
nobr, wbr, center,
centerOpen, centerClose,
dfn, big, small, sub, sup,
basefont,
fontOpen, fontClose,
plaintext, s, strike

– *List Tags*:

  `listHeader, listItem,`
  `ulistOpen, ulistClose,`
  `olistOpen, olistClose,`
  `dlistOpen, dlistClose,`
  `dlistTerm, dlistDef,`
  `menulistOpen, menulistClose,`
  `dirlistOpen, dirlistClose`

– *Character Format Tags*:

  `cite, code, emphasis, em,`
  `keyboard,`
  `kbd, sample, strong, variable`

– *Physical Format Tags*:

  `bold, italic, teletype`

– *Table Tags*:

  `tableOpen, tableRowOpen,`
  `tableHeader,`
  `tableData, tableCaption,`
  `tableRowClose,`
  `tableClose`

– *Form Tags*:

```
formOpen, formClose,
formCheckBox,
formHidden, formImage,
formPassword,
formRadio, formSubmit,
formReset,
formText, formSelectOpen,
formSelectOption,
formSelectClose,
formTextarea,
formTextareaOpen,
formTextareaClose,
formTextarea2,
formTextareaOpen2
```

- `HTF`. A hypertext function returns the HTML tag that corresponds to its name. However, it is not sufficient to call an HTF function on its own because the HTML tag is not passed to the PL/SQL Agent. The output of an HTF function must be passed to `htp.print` in order to actually be part of an HTML document. Every hypertext function (HTF) has a corresponding hypertext procedure (HTP) with the same name, parameters, and

function. HTF functions are generally used only when you need to nest calls.

- `OWA_UTIL`. This is a collection of useful utility procedures and functions. It is divided into the following areas:
  - `OWA_UTIL` HTML Utilities. The purposes of these range from printing a signature tag on HTML pages to retrieving the values of CGI environment variables and performing URL redirects.
  - `OWA_UTIL` Dynamic SQL Utilities. These enable the user to produce Web pages with dynamically generated SQL code.
  - `OWA_UTIL` Date Utilites. These make it easier to properly handle dates, which are simple strings in HTML, but are properly treated as a datatype by the Oracle RDBMS.
- `OWA_OPT_LOCK`. The procedures in this package enable the user to impose optimistic locking strategies so as to prevent lost updates.
- `OWA_PATTERN`. This is a set of procedures and functions the user can use to perform string

matching and substitution with rich regular expression functionality.

- `OWA_TEXT`. This is a set of procedures, functions, and datatypes used by `OWA_PATTERN` for manipulating large data strings. They are externalized so you can use them directly if you wish.
- `OWA_IMAGE`. This is a set of datatypes and functions for manipulating HTML image maps.
- `OWA_COOKIES`. This is a set of datatypes, procedures, and functions for manipulating HTML cookies.