A Simple Example

```
import oracle.html.*;
public class simple {
  public static void main (String args[]) {
    HtmlHead hd = new HtmlHead("Simple Example");
    HtmlBody bd = new HtmlBody();
    HtmlPage hp = new HtmlPage(hd, bd);
    hp.printHeader();
    bd.addItem(new SimpleItem(
       "This is a Heading (level2)").
       setHeading(2));
    bd.addItem(new SimpleItem("Good Day!").setBold())
      .addItem(SimpleItem.LineBreak);
    bd.addItem(SimpleItem.HorizontalRule);
    for (int i = 0; i < 5; i++) {
      bd.addItem(new SimpleItem("This is Line ")
                     .setEmphasis())
        .addItem(new SimpleItem(i))
        .addItem(SimpleItem.LineBreak);
    }
    bd.addItem(SimpleItem.HorizontalRule);
    Preformat pre = new Preformat();
    pre.addItem("void main () {\n")
       .addItem("  int i;\n\n")
       .addItem("  i = 5;\n")
       .addItem("  printf(\"i = %d\\n\",i);\n")
       .addItem("}\n");
    bd.addItem(pre);
    bd.addItem(SimpleItem.HorizontalRule);
    hp.print();
  } // end of main
} // end of class simple
```

The general procedure to create a dynamic HTML document is to create the following objects:

```
  HtmlHead hd = new HtmlHead("Document Title");
  HtmlBody bd = new HtmlBody();
  HtmlPage hp = new HtmlPage(hd, bd);
```

and use the `addItem` method to add HTML elements to the `bd` object. Finally, when the page is ready to be printed, the following statements can be used:

```
hp.printHeader();
hp.print()
```

The `SimpleItem` object is used to create HTML lines of text. It is a sub-class of the more general class `Item`. `SimpleItem` has three variables:

```
SimpleItem.Paragraph;
SimpleItem.LineBreak;
SimpleItem.HorizontalRule;
```

which can be used to generate the paragraph, line break and horizontal rule HTML elements respectively. There are numerous constructors for the `SimpleItem` object, which take as input objects of most common data types, including `string`, `int`, and `float`. Many of the `Item` class methods can be applied to `SimpleItem` objects such as `setBold()`, `setEmphasis()` etc.

HTML Form Processing

The Java Web Toolkit provides the class

```
public class Form extends CompoundItem
```

for HTML form processing. Since the `Form` class has been defined as a sub-class of `CompoundItem`, the `addItem` method can be applied to a `Form` object to populate it with various objects of the following types:

- `CheckBox`
- `Radio`
- `Submit`
- `Reset`
- `TextField`
- `TextArea`
- `Select`
- `Hidden`
- `PasswordField`

To create a Form object, the following steps must be taken:

- Create a form object, specifying the METHOD attribute (`GET` or `POST`), and the ACTION URL attribute.
- Populate the form object by creating one or more of the objects to be included in the form and add it to the form using the `addItem` method.
- Add the form object to the `HtmlBody` object.

Form creation – an example

```
import oracle.html.*;
public class params1 {
  public static void main (String args[]) {
    HtmlHead hd = new HtmlHead("Form Example");
    HtmlBody bd = new HtmlBody();
    HtmlPage hp = new HtmlPage(hd, bd);
    hp.printHeader();

    Form form1 = new Form("POST",
      "http://tinman.cs.gsu.edu:9001/book/"+
      "java/run/params2");

    Select select1 = new Select("sel");
    select1.addOption(new Option("One"));
    select1.addOption(new Option("Two"));
    select1.addOption(new Option("Three"));
    select1.addOption(new Option("Four"));
    form1.addItem(select1);
    Submit submit1 = new Submit("opt","Option1");
    Submit submit2 = new Submit("opt","Option2");
    bd.addItem(SimpleItem.LineBreak);
    CheckBox cb1 = new CheckBox("chk", "1111");
    CheckBox cb2 = new CheckBox("chk", "2222");
    CheckBox cb3 = new CheckBox("chk", "3333");
    CheckBox cb4 = new CheckBox("chk", "4444");
    CheckBox cb5 = new CheckBox("chk", "5555");
    CheckBox cb6 = new CheckBox("chk", "6666");
    CheckBox cb7 = new CheckBox("chk", "7777");
    form1.addItem(SimpleItem.HorizontalRule);
    form1.addItem(cb1); form1.addItem("1111");
    form1.addItem(SimpleItem.LineBreak);
    form1.addItem(cb2); form1.addItem("2222");
    form1.addItem(SimpleItem.LineBreak);
    form1.addItem(cb3); form1.addItem("3333");
    form1.addItem(SimpleItem.LineBreak);
    form1.addItem(cb4); form1.addItem("4444");
    form1.addItem(SimpleItem.LineBreak);
    form1.addItem(cb5); form1.addItem("5555");
    form1.addItem(SimpleItem.LineBreak);
    form1.addItem(cb6); form1.addItem("6666");
    form1.addItem(SimpleItem.LineBreak);
    form1.addItem(cb7); form1.addItem("7777");
    form1.addItem(SimpleItem.LineBreak);
    form1.addItem(submit1);
    form1.addItem(submit2);
    form1.addItem(new Hidden("dummy_ending","dummy"));
    bd.addItem(form1);

    hp.print();
  } // end of main
} // end of class params1
```

Retrieving form parameters

The Java program that receives the form parameters should have the following **main** method declaration:

```
public static void main(String args[]) {
...
}
```

The **args** array of **String** objects will contain the parameters sent from the Java program containing the form declaration. The individual strings in this array consists of the name/value pairs as follows:

```
args[i] = "parameter_name=value"
```

and the following code should extract the parameter value:

```
if (args[i].length() != 0) {
  if (args[i].startsWith("parameter_name",0)) {
   int pos = args[i].indexOf("=");
   parameter_value = args[i].substring(pos + 1);
  }
}
```

Multivalued form parameters

In contrast to the PL/SQL agent method, in which multi-valued form parameters were received in an array of values, multi-valued parameters in the Java Cartridge method are sent as multiple name/value pairs, where the name is the same in all the pairs. For example, if a select list called `sel1`, had three values, `val1`, `val2`, and `val3` associated with it, then the name/value pairs:

```
sel1=val1
sel1=val2
sel1=val3
```

will be sent to the form action procedure.

Processing form parameters

```
import oracle.html.*;
public class params2 {
  public static void main(String args[]) {
    HtmlHead hd = new HtmlHead("Main Page Output");
    HtmlBody bd = new HtmlBody();
    HtmlPage hp = new HtmlPage(hd, bd);
    hp.printHeader();

    String parameter1 = getArgument(args,"sel");
    hp.addItem(new SimpleItem(parameter1))
      .addItem(SimpleItem.Paragraph);

    for (int i=0; i < args.length; i++) {
      if (args[i].startsWith("chk")) {
        String parameter2 = args[i].substring(4);
        hp.addItem(new SimpleItem(parameter2))
          .addItem(SimpleItem.Paragraph);
      }
    }

    String parameter3 = getArgument(args,"opt");
    hp.addItem(new SimpleItem(parameter3))
      .addItem(SimpleItem.Paragraph);

    hp.print();
  }
} // end of class params2
```

```
private static String getArgument(String args[],
                                  String name) {
  String prefix = name + "=";
  for(int i = 0; i < args.length; i++)
    if (args[i].startsWith(prefix))
      return args[i].substring(prefix.length());
  return null;
}
```

Formatting HTML Tables

In order to create a table, the following steps must be taken:

1. Create a `DynamicTable` object.
2. Create a `TableRow` object for each row in the table.
3. Create either a `TableDataCell` object or a `TableHeaderCell` object for each cell in a row.
4. Add cells to each row by using the `addCell()` method.
5. Add all rows to the table by using the `addRow()` method.

```
import oracle.html.*;
public class dtable {
  public static void main (String args[]) {
    int MAXNUM = 10;

    HtmlHead hd = new HtmlHead("Dynamic Table Demo");
    HtmlBody bd = new HtmlBody();
    HtmlPage hp = new HtmlPage(hd, bd);
    hp.printHeader();

    DynamicTable tab = new DynamicTable(2);
    TableRow rows[] = new TableRow[MAXNUM];

    rows[0] = new TableRow();
    rows[0].addCell(new TableHeaderCell("I"))
           .addCell(new TableHeaderCell("I-SQUARED"));
    tab.addRow(rows[0]);
    for (int i=1; i< MAXNUM; i++) {
      rows[i] = new TableRow();
      rows[i].addCell(new TableDataCell(""+i))
             .addCell(new TableDataCell(""+i*i));
      tab.addRow(rows[i]);
    }

    bd.addItem(tab);
    bd.addItem(SimpleItem.HorizontalRule);
    hp.print();
  } // end of main
} // end of class dtable
```

Application – SQL Query Processor

```
import oracle.html.*;  // Java Cartridge Toolkit classes
import java.sql.*;     // JDBC classes
public class app1jdbc {
  public static void main (String args[])
        throws SQLException {
    HtmlHead hd = new HtmlHead("JavaApp Main Page");
    HtmlBody bd = new HtmlBody();
    HtmlPage hp = new HtmlPage(hd, bd);
    hp.printHeader();
    Form form1 = new Form("GET",
              "http://tinman.cs.gsu.edu:9001/"+
              "book2/java/run/app2jdbc");

    try {
      Class.forName ("oracle.jdbc.driver.OracleDriver");
    } catch (ClassNotFoundException e) {
        bd.addItem(
          new SimpleItem("Could not load the driver"));
        hp.print();
        return;
      }
    Connection conn = DriverManager.getConnection
                      ("jdbc:oracle:oci7:book/book");
    Statement stmt = conn.createStatement ();
    String query = "select db_name from db_list";
    ResultSet rset = stmt.executeQuery(query);

    Select select1 = new Select("dbname");
    while (rset.next ()) {
      select1.addOption(new Option(rset.getString(1)));
    }
    form1.addItem(select1);

    Submit submit1 = new Submit("submit1",
                                "Select_A_Database");
    form1.addItem(submit1);

    bd.addItem(form1);
    conn.close();
    hp.print();
  } // end of main
} // end of class app1jdbc
```

```
import oracle.html.*;
import java.sql.*;
public class app2jdbc {
  public static void main (String args[])
          throws SQLException {
    HtmlHead hd = new HtmlHead("SQL Query Page");
    HtmlBody bd = new HtmlBody();
    HtmlPage hp = new HtmlPage(hd, bd);
    hp.printHeader();

    String sDbname;
    sDbname = getArgument(args,"dbname");
    bd.addItem(new SimpleItem(
                  "Database Schema for ").setBold());
    bd.addItem(new SimpleItem(sDbname).setBold());
    bd.addItem(SimpleItem.LineBreak);
    bd.addItem(SimpleItem.HorizontalRule);
    try {
      Class.forName ("oracle.jdbc.driver.OracleDriver");
    } catch (ClassNotFoundException e) {
        bd.addItem(new SimpleItem(
              "Could not load the driver"));
        hp.print();
        return;
      }
    Connection conn = DriverManager.getConnection
                      ("jdbc:oracle:oci7:book/book");
    Statement stmt = conn.createStatement ();
    String query =
       "select table_name, column_name, data_type " +
       "from db_schema where db_name = ’" + sDbname + "’";
    ResultSet rset = stmt.executeQuery(query);
```

```
Form form1 = new Form("POST",
        "http://tinman.cs.gsu.edu:9001/book2/"+
        "java/run/app3jdbcmeta");

boolean newEntry = false;
String prev = new String();
int i=0;
while (rset.next ()) {
  if (i == 0) {
    bd.addItem(new SimpleItem(rset.getString(1)));
    bd.addItem(new SimpleItem("("));
    prev = new String(rset.getString(1));
  }
  else if (!prev.equals(rset.getString(1))) {
        bd.addItem(new SimpleItem(");"));
        bd.addItem(SimpleItem.LineBreak);
        bd.addItem(new SimpleItem(rset.getString(1)));
        bd.addItem(new SimpleItem("("));
        prev = new String(rset.getString(1));
        newEntry = false;
       }
  if (newEntry)
    bd.addItem(new SimpleItem(", "));
  newEntry = true;
  bd.addItem(new SimpleItem(rset.getString(2)));
  bd.addItem(new SimpleItem(":"));
  bd.addItem(new SimpleItem(rset.getString(3)));
  i++;
}
bd.addItem(new SimpleItem(");"));
bd.addItem(SimpleItem.LineBreak);
bd.addItem(SimpleItem.HorizontalRule);
```

```
TextField tf1 = new TextField(
                    "select_clause",100,50,"");
TextField tf2 = new TextField("from_clause",100,50,"");
TextArea ta1 = new TextArea("where_clause",50,10);

DynamicTable tab = new DynamicTable(2);
TableRow rows[] = new TableRow[3];
rows[0] = new TableRow();
rows[0].addCell(new TableDataCell(new SimpleItem(
                       "SELECT").setBold()))
       .addCell(new TableDataCell(tf1));
tab.addRow(rows[0]);
rows[1] = new TableRow();
rows[1].addCell(new TableDataCell(new SimpleItem(
                       "FROM").setBold()))
       .addCell(new TableDataCell(tf2));
tab.addRow(rows[1]);
rows[2] = new TableRow();
rows[2].addCell(new TableDataCell(new SimpleItem(
                       "WHERE").setBold()))
       .addCell(new TableDataCell(ta1));
tab.addRow(rows[2]);
```

```
    form1.addItem(tab);

    Hidden hdd = new Hidden("dbname",sDbname);
    form1.addItem(hdd);

    Submit submit1 = new Submit();
    form1.addItem(submit1);

    Reset reset1 = new Reset();
    form1.addItem(reset1);

    bd.addItem(form1);
    conn.close();
    hp.print();

  }
} // end of class app2jdbc
```

```
import oracle.html.*;
import java.util.*;
import java.sql.*;
public class app3jdbcmeta {
  public static void main (String args[])
        throws SQLException, Exception {
    HtmlHead hd = new HtmlHead("Query Results Page");
    HtmlBody bd = new HtmlBody();
    HtmlPage hp = new HtmlPage(hd, bd);
    hp.printHeader();
    String select_string;
    String from_string;
    String where_string_temp;
    String where_string;
    String sDbname;
    sDbname = getArgument(args,"dbname");
    select_string = getArgument(args,"select_clause");
    from_string = getArgument(args,"from_clause");

    where_string_temp = getArgument(args,"where_clause");
    int nn;
    StringBuffer sb = new StringBuffer(
                            where_string_temp.length());
    for (int k = 0; k < where_string_temp.length(); k++) {
        nn = (int) where_string_temp.charAt(k);
        if ((nn == 13) || (nn == 10))
          sb.insert(k,(char) 32);
        else
          sb.insert(k,where_string_temp.charAt(k));
    }
    where_string = sb.toString();
```

```
// construct entire query from
// select_string,from_string, where_string
String query;
if (where_string.length() == 0)
  query = "select distinct " + select_string +
                 " from " + from_string;
else
  query = "select distinct " + select_string +
                 " from " + from_string +
                 " where " + where_string;
bd.addItem(new SimpleItem("Query is: "))
  .addItem(new SimpleItem(query))
  .addItem(SimpleItem.Paragraph);
int nCols;
Link link1,link2;
try {
  Class.forName ("oracle.jdbc.driver.OracleDriver");
} catch (ClassNotFoundException e) {
    bd.addItem(new SimpleItem("Error"));
    hp.print();
    return;
  }
Connection conn = DriverManager.getConnection
                  ("jdbc:oracle:oci7:book/book");
Statement stmt = conn.createStatement ();
```

```
ResultSet rset;
ResultSetMetaData rsetmd;
try {
  rset = stmt.executeQuery(query);
  rsetmd = rset.getMetaData();
} catch (SQLException e) {
    bd.addItem(new SimpleItem("Query not OK"))
      .addItem(SimpleItem.Paragraph);
    bd.addItem(SimpleItem.HorizontalRule)
      .addItem(SimpleItem.Paragraph);
    link1 = new Link (
        "http://tinman.cs.gsu.edu:9001/book2/"+
        "java/run/app2jdbc?dbname="
        +sDbname, "Another Query?");
    link2 = new Link (
        "http://tinman.cs.gsu.edu:9001/book2/"+
        "java/run/app1jdbc",
        "Query Another Database?");
    bd.addItem(link1).addItem(SimpleItem.Paragraph);
    bd.addItem(link2).addItem(SimpleItem.Paragraph);
    hp.print();
    return;
  }
```

```
nCols = rsetmd.getColumnCount();
DynamicTable tab = new DynamicTable(nCols);
tab.setBorder(2);
TableRow rows[] = new TableRow[1000];
rows[0] = new TableRow();
for (int i1 = 1; i1 <= nCols; i1++) {
  rows[0].addCell(new TableHeaderCell(
      new SimpleItem(rsetmd.getColumnName(i1))));
}
tab.addRow(rows[0]);
```

```
int nRows = 0;
while (rset.next ()) {
  rows[nRows] = new TableRow();
  for (int c=1; c <= nCols; c++) {
    rows[nRows].addCell(new TableDataCell(
          new SimpleItem(rset.getString(c))));
  }
  tab.addRow(rows[nRows]);
  nRows++;
}

if (nRows > 0) {
  bd.addItem(tab);
  bd.addItem(SimpleItem.Paragraph);
  bd.addItem(new SimpleItem("Number of rows="))
    .addItem(new SimpleItem(nRows))
    .addItem(SimpleItem.Paragraph);
  bd.addItem(SimpleItem.HorizontalRule)
    .addItem(SimpleItem.Paragraph);
  link1 = new Link (
     "http://tinman.cs.gsu.edu:9001/book2/java/"+
     "run/app2jdbc?dbname="
     +sDbname, "Another Query?");
  link2 = new Link (
     "http://tinman.cs.gsu.edu:9001/book2/java/"+
     "run/app1jdbc",
     "Query Another Database?");
  bd.addItem(link1).addItem(SimpleItem.Paragraph);
  bd.addItem(link2).addItem(SimpleItem.Paragraph);
}
```

```
    if (nRows == 0) {
      bd.addItem(new SimpleItem("No rows in result"))
        .addItem(SimpleItem.Paragraph);
      bd.addItem(SimpleItem.HorizontalRule)
        .addItem(SimpleItem.Paragraph);
      link1 = new Link (
          "http://tinman.cs.gsu.edu:9001/book2/java/"+
          "run/app2jdbc?dbname="
          +sDbname, "Another Query?");
      link2 = new Link (
          "http://tinman.cs.gsu.edu:9001/book2/java/"+
          "run/app1jdbc",
          "Query Another Database?");
      bd.addItem(link1).addItem(SimpleItem.Paragraph);
      bd.addItem(link2).addItem(SimpleItem.Paragraph);
    }
    hp.print();
  }
} // end of class app3jdbcmeta
```