Part 4: Creating a Drill-down Interface

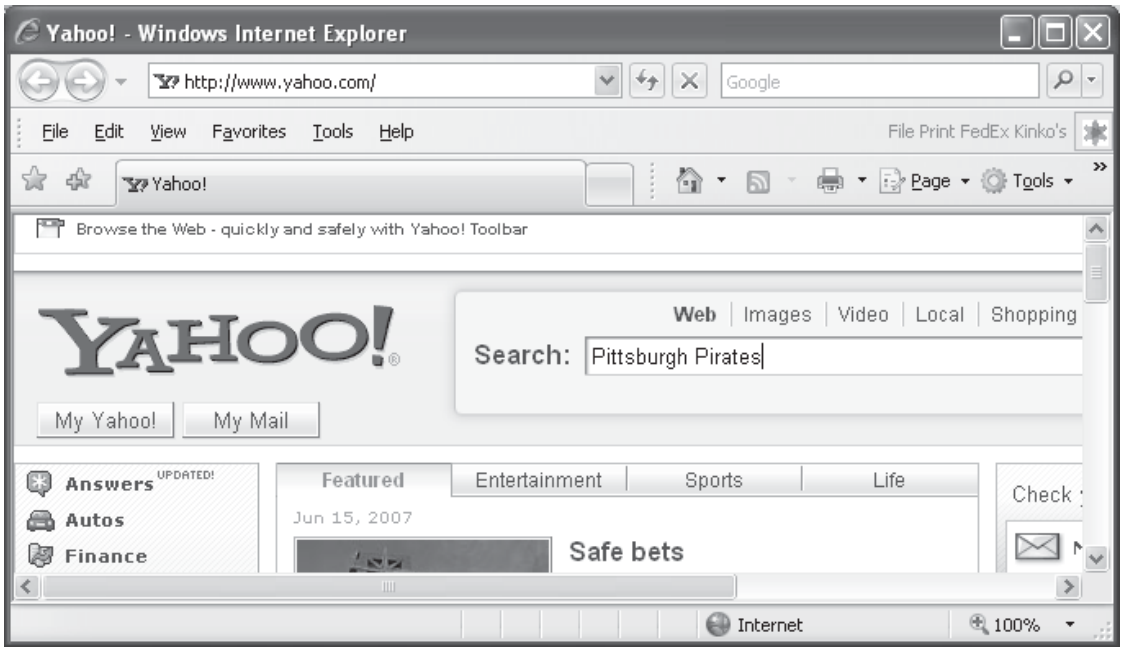
In this section you will learn to

- Pass Variables through a link
- Receive Variables passed through the URL

Passing Variables between Pages

Variables may be passed between pages in two main ways: through links and through HTML forms.

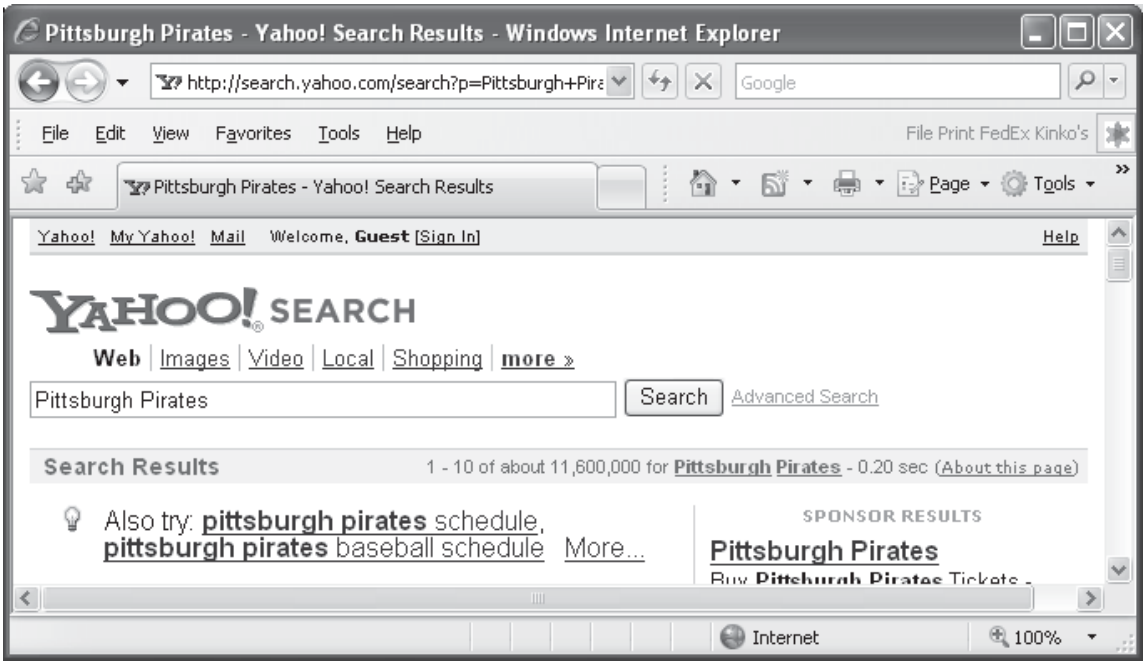
Consider the following example. At http://www.yahoo.com, perhaps the most important element on the page is a text field with a submit button that says "Search."



If the user types in "Pittsburgh Pirates" in that search box and presses submit, the user is taken to another page:

http://search.yahoo.com/bin/search?**p=Pittsburgh+Pirates**

A look at the source code of the first page will reveal that the text field is named “p”. The second page is written to accept this value and uses it to search through Yahoo’s database of websites. The resulting records are then displayed.



“Two-page method”

This system explained above is called the “two-page method.” There are two pages required for the complete system to work. If a user attempts to visit the second page without passing the required variables, errors may occur. We will have to build systems that anticipate what users will do and will protect the site from errors.

Later in the course we will examine a more complicated “one-page method” where a page passes a variable to itself. It is only more complicated in the sense that all code is held on one page. While the one-page method can lead to a long page that needs to be organized well, it does have other benefits that will be seen later.

Demo: Pass Variables via Links

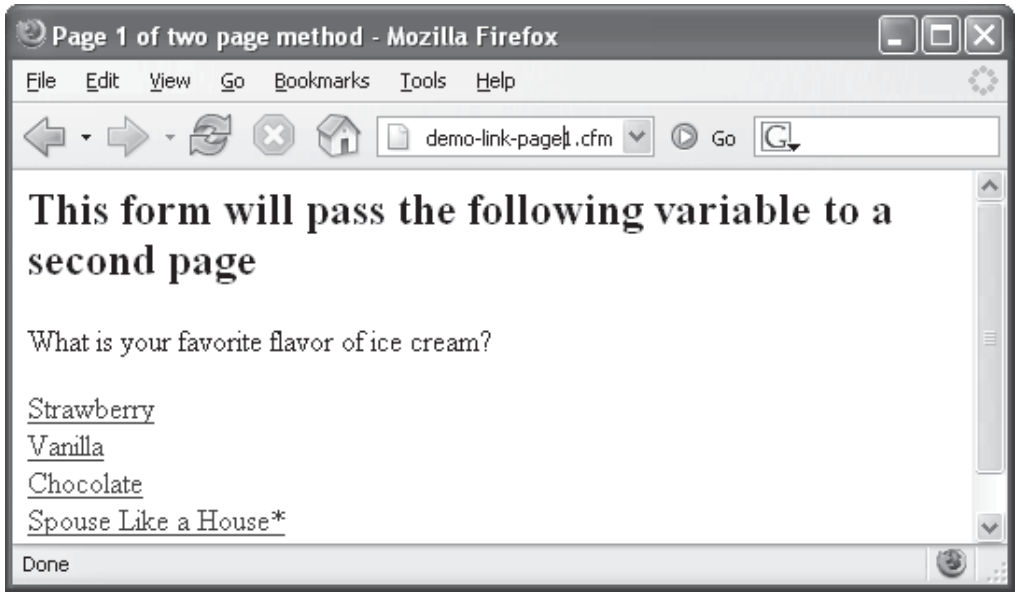
Forms will be covered in more detail in the next section, for now, just focus at the URL. Immediately following the name of the page is a question mark and some data in the form of a “name=value” pair.

It is also possible to pass variables through links. In fact, look above at the URL that was generated when the submit button was clicked. It should look something like this:

http://search.yahoo.com/bin/search?**p=Pittsburgh+Pirates**

Immediately after the name of the page is a question mark followed by a variable name-value pair. This name-value pair is available to the second page for display, for storage in a database, or for mathematical equations.

The same type of name-value pairs can be passed without a form by passing the information inside a link. The entire URL above can be used inside the href value of a link. You will find the following demo saved as **demos/demo-link-page1.cfm**.



The code of the above page is below. Note the value of the href attributes:

```
<html>
<head>
   <title>Page 1 of two page method</title>
</head>

<body>
<h2>This form will pass the following variable to a second page</h2>

<p>What is your favorite flavor of ice cream?</p>

<a href="demo-link-page2.cfm?flavor=Strawberry">Strawberry</a><br>
<a href="demo-link-page2.cfm?flavor=Vanilla">Vanilla</a><br>
<a href="demo-link-page2.cfm?flavor=Chocolate">Chocolate</a><br>
<a href="demo-link-page2.cfm?flavor=Spouse+Like+a+House">Spouse Like a
House *</a><br>
<a href="http://www.handelsicecream.com/">* Handel's Ice Cream - Yum</a>

</body>
</html>
```

Demo: List of Actors with links

The following page, saved as **solutions/allactors.cfm** shows a list of actors where each actor's name has been turned into a link using the following form:

```
actorDetails.cfm?actor_id=10
```

In the following several pages we will receive and process the variables passed through the URL.

Receive Variables through the URL

Now that you know how to pass a variable through the URL, you must learn how to receive it on the second page.

The name “drill-down” comes from the fact that we are getting more information or “drilling down for information.” It generally means that we are passing something like an id number to the second page which will query the database for some more detail on that item (perhaps additional product information).

Using a WHERE clause in a SQL Statement

All of our SQL statements so far have been selecting all records in the database. This type of query does not require a WHERE clause (although it may have one). The WHERE clause filters the records. In this section we will add a filter that will force the query to return exactly one record. It will then provide more details on that record than were available in the previous page.

The demo below is saved as **demos/demo-actorDetails.cfm**, however, it is not accessible directly, and it must receive an actor_id. That value is passed from the actors select list demonstrated in the last section. Look at the URL in the following screenshot. The actor_id is visible. It was sent from the link on **demos/allactors.cfm**.

Once it receives the actor_id, this page queries the database and brings back any matching records. In this case, that must be exactly one record because we are searching on the primary key.

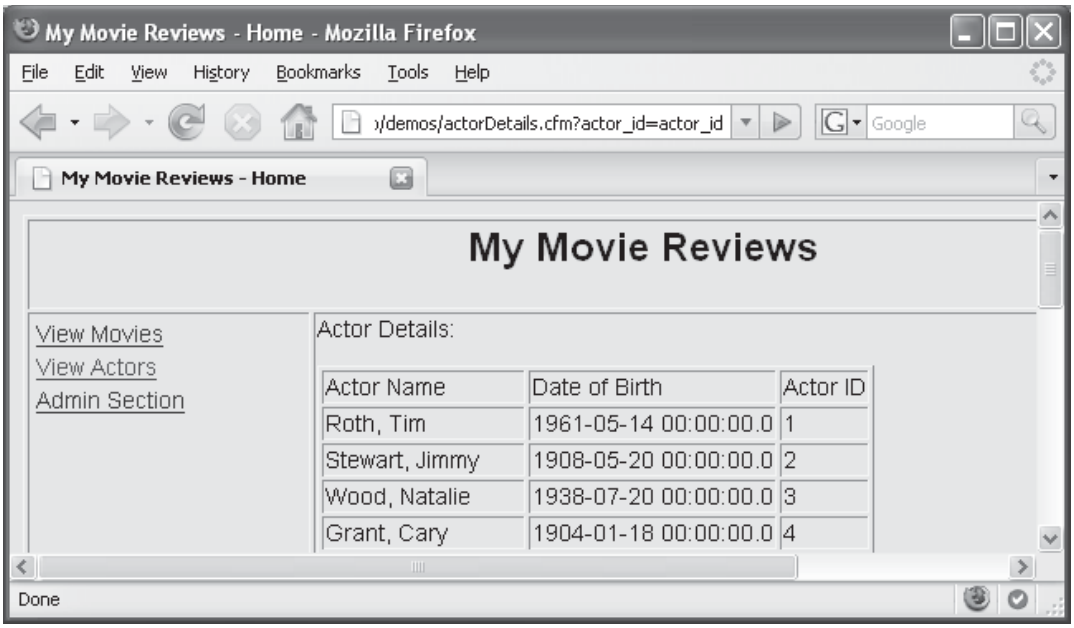
```
<cfquery name="getActors" datasource="movieList">
   SELECT *
   FROM actors
   WHERE actor_id = #URL.actor_id#
</cfquery>
<table border="1">
  <tr>
    <td>Actor Name</td>
    <td>Date of Birth</td>
    <td>Actor ID</td>
  </tr>
  <cfoutput query="getActors">
    <tr>
      <td>#getActors.lastname#, #getActors.firstname#</td>
      <td>#getActors.dob#</td>
      <td>#getActors.actor_id#</td>
    </tr>
  </cfoutput>
</table>
```

Protecting Against SQL Injection Attacks with <cfqueryparam>

When passing variables from page to page, you are always vulnerable to a malicious hacker who might try to change the query string. At worst, the hacker could attempt to change or even delete your database. (Read much more about SQL Injection Attacks in the Wikipedia entry at: http://en.wikipedia.org/wiki/SQL_Injection.)

In the actor details page shown above, by simply changing the URL from http://localhost/ColdFusionIntro/demos/actorDetails.cfm?actor_id=**1** to http://localhost/ColdFusionIntro/demos/actorDetails.cfm?actor_id=**actor_id** would instead display ALL actor's detail information!

The screenshot below shows multiple actors' information.



One simple tool ColdFusion provides to prevent these problems is the <cfqueryparam> tag. It performs some simple validation on a parameter before it is used in a query. Compare the two queries below:

```
<cfquery name="getActors" datasource="movieList">
   SELECT *
   FROM actors
   WHERE actor_id = #URL.actor_id#
</cfquery>

<cfquery name="getActors" datasource="movieList">
   SELECT *
   FROM actors
   WHERE actor_id = <cfqueryparam value="#URL.actor_id#"
cfsqltype="CF_SQL_INTEGER">
</cfquery>
```

This code is saved as **demos\allactors-queryparam.cfm** and **demos\allactors-queryparam.cfm**.

It is recommended that you use <cfqueryparam> in every <cfquery> that uses parameters.

Exercise 4: Drill-down for More Detail

15 to 20 minutes

In this exercise you will add links to the movie list. When the user clicks on the link, the movie's ID will be passed to a details page which will show additional information about the movie.

Part 1: Add the links to the allmovies.cfm page.

1. Open **allmovies.cfm** (the file from the last two exercises).

2. Wrap an <a> tag around the movie name. Include the movie_id so that it is passed to a page called movieDetails.cfm as shown below:

```
<a href="movieDetails.cfm?movie_id=#movie_id#">#name#</a>
```

3. You may test this page in the browser now, but don't bother to click on the links. The second page has not yet been finished. When you are done, the first page will look something like this:

Part 2: Build the detail page.

1. Open **movieDetails.cfm**.

2. Find the following comment in the main content table cell.

```
<!---
  1 - Add a query that uses a WHERE clause which will
  pull the movie_id from the URL
  2 - Add a table to display the movie data including:
  name, genre, summary, release_year
--->
```

3. Beneath the comment add a query that searches the database for the record that matches the movie_id in the URL.

4. Add an output block that shows any fields from the movie table you would like to display. Below you will see the following fields: name, genre, summary, release_year.

5. To test, begin on allmovies.cfm. Click on a link to pass a movie_id to the second page. When you are done, it might look something like this:

Challenge

- What happens if the user goes directly to this page without passing the appropriate variable? Add a conditional statement that prevents that error. This is covered in the next section.

Possible Solution to Exercise 4

Saved as **solutions /allmovies-ex-04-done.cfm**

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>My Movie Reviews - Home</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link href="mymovies.css" rel="stylesheet" type="text/css">
</head>

<body>
<table width="750" border="1">
  <tr>
    <td colspan="2" valign="top"> <h2 align="center">My Movie
Reviews</h2>
      </td>
  </tr>
  <tr valign="top">
    <td width="175" valign="top">

   <!--- Navigation Table --->
   <table width="100%" border="0">
        <tr>
        <td><a href="allmovies.cfm">View Movies</a></td>
        </tr>
        <tr>
          <td><a href="allactors.cfm">View Actors</a></td>
        </tr>
        <tr>
          <td><a href="admin.cfm">Admin Section</a></td>
        </tr>
        <tr>
          <td> </td>
        </tr>
      </table>

    </td>
    <td><p>Movie List:</p>
      <cfquery name="getMovies" datasource="movieList">
         SELECT name, movie_id, genre
         FROM movies
         ORDER BY name
      </cfquery>

     <table width="100%" border="1">
        <tr>
          <td>Movie Name </td>
          <td>Genre</td>
          <td>Movie ID </td>
        </tr>
```

```
       <cfoutput query="getMovies"> <tr>
          <td><a
href="movieDetails.cfm?movie_id=#movie_id#">#name#</a></td>
          <td>#genre#</td>
          <td>#movie_id#</td>
        </tr></cfoutput>
      </table>
     <p>  </p>
      <p> </p>
      <p> </p>
      <p> </p></td>
  </tr>
  <tr>
    <td colspan="2" valign="top">
   <cfinclude template="footer-date.cfm">
   </td>
  </tr>
</table>

</body>
</html>
```

Saved as **solutions /movieDetails-ex-04-done.cfm**

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>My Movie Reviews - Home</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link href="mymovies.css" rel="stylesheet" type="text/css">
</head>

<body>
<table width="750" border="1">
  <tr>
    <td colspan="2" valign="top"> <h2 align="center">My Movie
Reviews</h2>
      </td>
  </tr>
  <tr valign="top">
    <td width="175" valign="top">

   <!--- Navigation Table --->
   <table width="100%" border="0">
        <tr>
        <td><a href="allmovies.cfm">View Movies</a></td>
        </tr>
        <tr>
          <td><a href="allactors.cfm">View Actors</a></td>
        </tr>
        <tr>
          <td><a href="admin.cfm">Admin Section</a></td>
        </tr>
        <tr>
          <td> </td>
```

```
        </tr>
      </table>

    </td>
    <td><p>Movie Details:</p>
      <cfquery name="getMovies" datasource="movieList">
         SELECT name, movie_id, genre, summary, release_year
         FROM movies
         WHERE movie_id = #URL.movie_id#
      </cfquery>

     <table width="100%" border="1">
         <cfoutput query="getMovies">
       <tr>
          <th>Movie Name </th>
          <td>#name#</td>
        </tr>
       <tr>
          <th>Genre</th>
          <td>#genre#</td>
        </tr>
      <tr>
          <th>Summary</th>
          <td>#summary#</td>
        </tr>
      <tr>
          <th>Release Year</th>
          <td>#release_year#</td>
        </tr>
      </cfoutput>
      </table>
     <p>  </p>
      <p> </p>
      <p> </p>
      <p> </p></td>
  </tr>
  <tr>
    <td colspan="2" valign="top">
   <cfinclude template="footer-date.cfm">
   </td>
  </tr>
</table>

</body>
</html>
```