24 - HTML for Forms Professionals

Track: Introductory / Field: IS

Instructor: Mr. Franklin J. Garner, III - President and CEO, Amgraf, Inc.

Day: Tuesday, May 23

Time: 8:00 am – 11:30 am – 3.0 hours

Amgraf, Inc.
1501 Oak Street
Kansas City, MO 64108
(816) 474-4797
www.amgraf.com

This Page Intentionally Left Blank

HTML for Forms Professionals

By Franklin J. Garner, III
President and CEO
Amgraf, Inc.

What is HTML?

HTML is an abbreviation for *Hypertext Markup Language*, and it is the primary language of the World Wide Web. Hypertext refers to *linked content*, generally defined as any text that contains links to other documents. The World Wide Web is different from prior information resources mainly because of hyperlinking, and the opportunity that linking offers for users to choose individual paths through content. HTML is used to specify (or markup) typographical attributes about the content of web pages, and web browsers interpret HTML to control the display of these pages. Although HTML is a language, and therefore has its own rules, it is much simpler than most other programming languages (such as C).

HTML documents are simply ASCII (text character) files that contain embedded formatting *tags*. These tags are shorthand codes that start with a less-than (<) character and end with a greater-than (>) character. Typically an HTML document includes formatting tags to specify headings, paragraphs, tables, centering, bold text, horizontal rules, and bulleted lists, etc. Hyperlinks are indicated by a tag called the *Anchor* tag. Images, logos, and photos are referenced with HTML tags that point to external source files. Most HTML tags also have optional *attributes* that extend their functionality. Below is a list of the most commonly used HTML 4.0 tags. *Note that HTML tags are not case-sensitive. They can be either upper or lower case.*

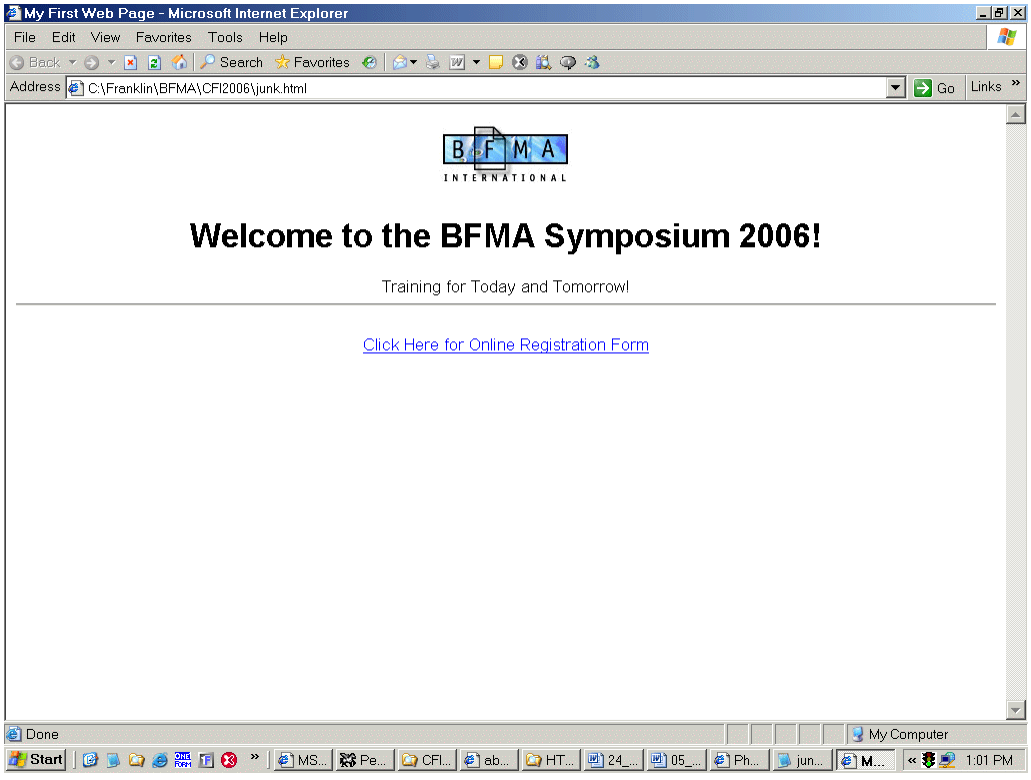
HTML Tag	Description
<!--...-->	Defines a comment
<a>	Defines an anchor
<b>	Defines bold text
<body>	Defines the body element
 	Inserts a single line break
	Defines centered text
	Defines a section in a document
<font>	Defines the font face, size, and color of text
<form>	Defines a form
<h1> to <h6>	Defines header 1 to header 6
<head>	Defines information about the document
<hr>	Defines a horizontal rule
<html>	Defines an html document
<i>	Defines italic text
<img>	Defines an image
<input>	Defines an input field
<label>	Defines a label for a form control
<meta>	Defines meta information
	Defines an ordered list
<option>	Defines an option in a drop-down list
<p>	Defines a paragraph
<select>	Defines a selectable list
<table>	Defines a table
<td>	Defines a table cell
<textarea>	Defines a text area
<title>	Defines the document title
<tr>	Defines a table row

Creating a Simple Web Page with HTML

Although most professional web page designers use HTML-editing software that automatically generates perfect HTML code from visual designs, it is useful to know some of the basic functions and syntax of the HTML language. You can quickly create a simple web page with a text editor like the Microsoft Windows Notepad. For example, the 12 lines of HTML code shown below:

```
<HTML>
<HEAD>
<TITLE> My First Web Page </TITLE>
</HEAD>
<BODY><CENTER>
<IMG SRC="bfma.jpg" WIDTH="133" HEIGHT="70" BORDER="0">
<H1>Welcome to the BFMA Symposium 2006!</H1>
<P>Training for Today and Tomorrow!
<HR>
<P><A HREF="form1.html">Click Here for Online Registration Form</A>
</CENTER></BODY>
</HTML>
```

display the page in my browser like this:



Note that HTML is intentionally a highly adaptive presentation technology. My HTML file may look very different on your computer, depending on your choices of screen fonts, text sizes, background colors and other settings within your browser. If the form designer wishes to override the user's browser settings, there are many HTML *attribute* tags that can be inserted into the code.

Decoding the HTML from the Example Page

From the example above you can see that it is easy to get started with HTML. Now let's break down what these HTML tags in our first web page mean to the browser.

Note that most HTML tags have a Begin state and an End state to indicate where an effect starts and stops. The syntax is as follows: <TAG> for start and </TAG> for stop. However, there are many exceptions to this rule.

- The first line in the file is the **<HTML>** tag. This tells the browser to expect an HTML-coded page to follow.
- The next line is the **<HEAD>** tag, which is a section at the beginning of an HTML file used to insert information about the page that DOES NOT display in the browser. The Head section is often used to identify the author, to include searchable terms, and for a page title.
- The next line is the **<TITLE>** tag and the words "My First Web Page" will be displayed in the title bar of the web page. Note that the **</TITLE>** tag defines where the title text ends.
- The next line closes the Head section with the **</HEAD>** tag.
- The **<BODY>** tag indicates the actual beginning of the web page content. The **<CENTER>** tag indicates that all following content is to be displayed horizontally centered on the web page.
- The **<IMG>** tag is the pointer to an external file that is the BFMA logo. Note that the IMG tag does not have a separate end tag.
- The **<H1>** tag specifies a large bold font for the main heading of a page. (H1-H6 is largest to smallest.)
- The **<P>** tag specifies a regular paragraph font for body text. Note that the </P> end paragraph tag is optional.
- The **<HR>** tag draws a medium gray horizontal rule across the web page.
- The **<P><A HREF=>** tags indicate a linked paragraph, and specify the Uniform Resource Locator (URL) to branch to when the link is clicked.
- The last two lines close the BODY, CENTER, and HTML states that were started earlier in the file.

HTML Form Basics

The HTML page shown in the above example is sometimes referred to as *static page*. To be useful, an HTML form must be an *interactive page*. Every web form needs to capture data keyed into fill-fields or selected from drop-down lists.

To build HTML forms, there are three forms-oriented HTML tags that create fill-fields and buttons. It is fairly simple to use the forms tags as the following example will show. However, in addition to the HTML form, you will need a *form handler*. A form handler is a web page server program that processes the submitted form data in some way, either by sending it via E-mail to you, or by storing it to a file or database table that you can access when needed. There are several popular commercial forms software packages that provide the server form handler functionality along with the HTML forms.

Since server-side programming is beyond the scope of this presentation, we will focus purely on the HTML form design issues and techniques.

Creating HTML Forms

Web pages are upgraded to HTML Forms by inserting the *Form*, *Input*, and *Select* tags in the file to display fields, dropdown lists, and buttons. Intelligent forms can also include logical operations like calculating a sum of two fields. These actions are possible because embedded JavaScript routines can be triggered by HTML *events*. An example is to execute a calculation routine when a user clicks on an HTML element.

Let's look at these HTML tags:

HTML Tag	Description
<form>	Defines a form
<input>	Defines an input field
<select>	Defines a selectable list

The **<FORM>** tag is used to indicate that the HTML page has the properties of an interactive form. In particular, this tag specifies the action and the method to be used to collect the data when it is submitted.

The **<INPUT>** tag is used for each fill-field and for each button on the form.

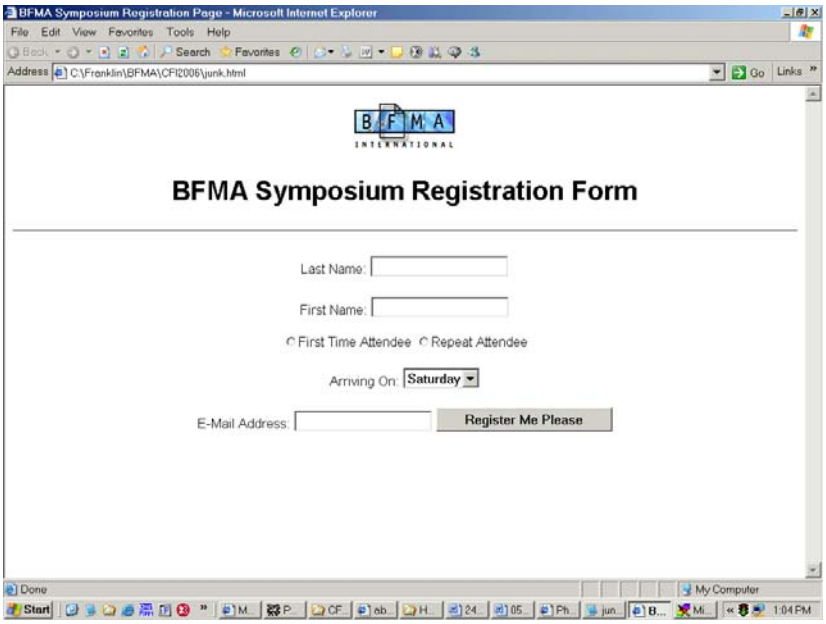
The **<SELECT>** tag is used when a pre-defined list of choices is to be presented to the client. This is sometimes called a *dropdown list*.

HTML Code for a Registration Form

Compared to static web pages, forms are slightly more complex to code in HTML. Following is a new HTML file that includes form fields and a submit button. The HTML tags for forms are in bold face:

```
<HTML>
<HEAD>
<TITLE> BFMA Symposium Registration Page </TITLE>
</HEAD>
<BODY><CENTER>
<IMG SRC="bfma.jpg" WIDTH="133" HEIGHT="70" BORDER="0">
<H1>BFMA Symposium Registration Form</H1>
<HR>
<FORM METHOD=POST ACTION="http://www.amgraf.com/emailxml/save_emailxml-pl.cgi">
<P>Last Name:
<INPUT TYPE="text" NAME="LastName">
<P>First Name:
<INPUT TYPE="text" NAME="FirstName"><BR><BR>
<INPUT TYPE="radio" NAME="BHDT" value="FTA"><LABEL>First Time Attendee</LABEL>
<INPUT TYPE="radio" NAME="BHDT" value="RA"><LABEL>Repeat Attendee</LABEL><BR>
<P>Arriving On:
<SELECT NAME="ArrivalDay">
<OPTION value="1">Saturday</OPTION>
<OPTION value="2">Sunday</OPTION>
<OPTION value="3">Monday</OPTION>
<OPTION value="4">Tuesday</OPTION>
</SELECT>
<P>E-Mail Address:
<INPUT TYPE="text" NAME="EmailAddr">
<INPUT TYPE="submit" value="Register Me Please">
</FORM>
</CENTER></BODY>
</HTML>
```

This HTML file displays in my browser as shown on the next page:



Decoding the HTML from the Registration Form Page

We already looked at the tags in a static HTML page. Now let's break down the additional HTML tags needed to create a fillable, submittable form.

- The **`<FORM METHOD=POST ACTION="http://www.amgraf.com/emailxml/save_emailxml-pl.cgi">`** line is the indicator to the browser that this page contains a form. The *Method* attribute indicates that the data from the fill-fields will be *Posted* to the server. This means that there will be *fieldname/data value* pairs (one pair for each field) transmitted to the server when the Submit button is clicked. The *Action* attribute indicates the URL of the program on the server that will process the form data.

- The **`<INPUT TYPE="text" NAME="LastName">`** line is the HTML code necessary to create the first fill-field. This is the field that will capture the registrant's last name. The field is to be used for keying textual information and the field name is "LastName". A second similar input field follows for the first name.

- The next line contains **`<INPUT TYPE="radio" NAME="BHDT" value="FTA"><LABEL>First Time Attendee</LABEL>`**. This is the code to create a field type called Radio Button. On this form there are two mutually exclusive radio buttons. Only one of the buttons can be on at a time. This action is controlled by duplicating the NAME="BHDT" attribute for this field and for the next field. The value returned to the server if this field is clicked on, will be "FTA", my notation for First Timer.

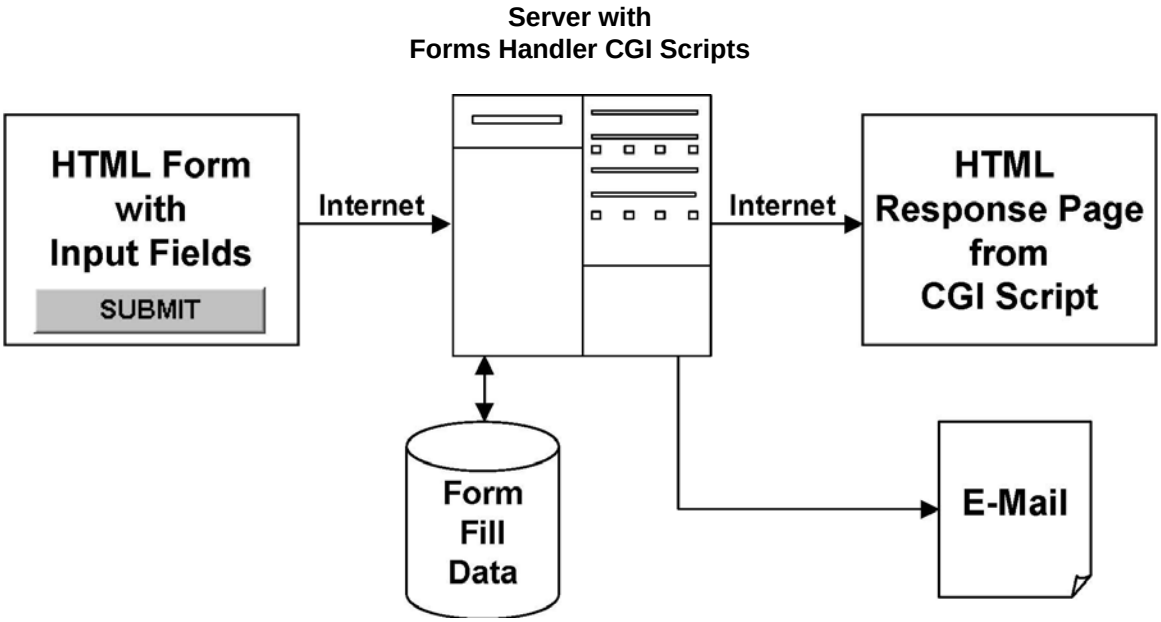
- To create the dropdown list, the following code is inserted. This causes a drop down list with 4 choices to be displayed. The field name is ArrivalDay, and the value submitted to the server is "1" if the choice is Tuesday, "2" if Wednesday, etc.

```
<SELECT NAME="ArrivalDay">
<OPTION value="1">Saturday</OPTION>
<OPTION value="2">Sunday</OPTION>
<OPTION value="3">Monday</OPTION>
<OPTION value="4">Tuesday</OPTION>
</SELECT>
```

- Finally, the form is submitted by clicking the Submit button drawn by the **`<INPUT TYPE="submit" value="Register Me Please">`** code. The file ends with the closing tags for **`</FORM> </CENTER> </BODY>`** and **`</HTML>`**.

Submitting and Testing the HTML Form

A connection to the Internet is all that is required to submit the form fill data. The *fieldname/data value* pairs (one pair for each field) will be transmitted to the forms handler on the web server when the Submit button is clicked. In this example, the form data will then be emailed to the address entered into the sample form. A more sophisticated system would also store the information into database tables, as shown in the following diagram.



Intermission

The class is adjourned for one-half hour at this point in the presentation. Intermediate HTML concepts will be discussed in the next session.

Intermediate HTML Forms Functionality

As explained in the beginning of this class, HTML is a content presentation language. HTML tags can be inserted into an ASCII file to specify typefaces, rule lines, and logos. Fillable forms are created by using the HTML *Form*, *Input*, and *Select* tags to draw fields, dropdown lists, and buttons.

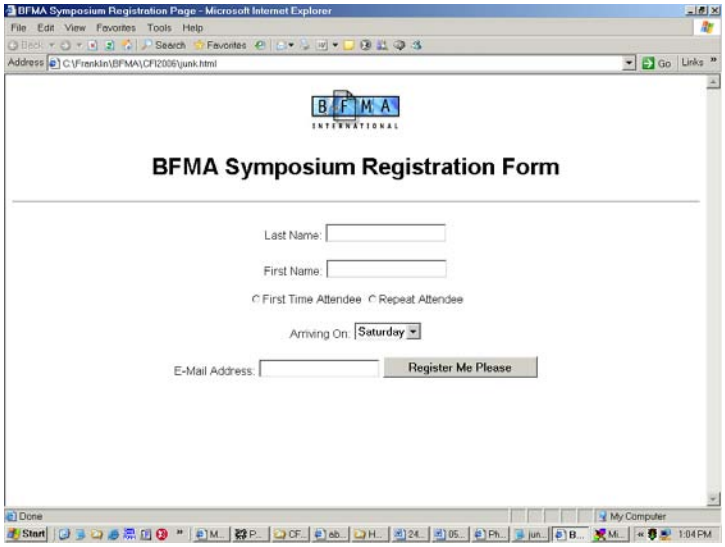
Intermediate HTML forms require an understanding of JavaScript. HTML is great for laying-out a form design, but the language unfortunately does not support arithmetic or logical operations. Intelligent forms often include logical operations like calculating a sum of two fields. For computation functions, most Internet forms designers therefore include JavaScript code inside the HTML form files.

JavaScript (not related to *Java*) is a scripting language introduced by Netscape as a general-purpose programming language for enhancing Web pages. JavaScript code is embedded as small programs inside a web page that are interpreted and executed by the Web client. The forms programmer determines the time and nature of the execution, often triggering JavaScript functions by mouse clicks, buttons, or other actions from the user. JavaScript functions (like subroutines) can also be called from within a Web document to fully control Web browsers, including all the familiar browser attributes.

In this session, I will demonstrate how to use Amgraf's OneForm Designer Plus (OFDP) HTML forms design software to create onscreen, fillable, database-connected, e-mailable Internet forms. I will "open the hood" to show the HTML and JavaScript coding that is produced by the forms design software. Keep in mind however that through the latest forms design tools, many functions and features can be implemented without needing to know the nuances of HTML, JavaScript, or web server scripts.

Enhancing HTML Forms with JavaScript

Following is the HTML registration form demonstrated in the *Introduction to HTML for Forms Professionals* class:



Now we will make an enhancement to the Last Name field using JavaScript.

Keystroke Validation Enhancement

Keystroke validation is a desirable feature in online forms because it can help prevent bad information from being submitted. Some of the common types of keystroke validation are Numeric Only, Uppercase Only, and counting characters.

Following is a printout of the HTML file for the Registration Form. HTML and JavaScript codes have been inserted to force the field named “LastName” to be all uppercase letters. The changes are highlighted with boldface.

```
<HTML>
<HEAD>
<TITLE> BFMA Symposium Registration Page </TITLE>
</HEAD>
<BODY><CENTER>
<IMG SRC="bfma.jpg" WIDTH="133" HEIGHT="70" BORDER="0">
<H1>Symposium Registration Form</H1>
<HR>
<FORM METHOD=POST ACTION="http://www.amgraf.com/emailxml/save_emailxml-pl.cgi">
<P>Last Name:
<INPUT TYPE="text" NAME="LastName" onKeyPress="keypress_LastName(event)">
First Name:
<INPUT TYPE="text" NAME="FirstName"><BR><BR>
<INPUT TYPE="radio" NAME="BHDT" value="FTA"><LABEL>First Time Attendee</LABEL>
<INPUT TYPE="radio" NAME="BHDT" value="RA"><LABEL>Repeat Attendee</LABEL><BR>
<P>Arriving On:
<SELECT NAME="ArrivalDay">
<OPTION value="1">Saturday</OPTION>
<OPTION value="2">Sunday</OPTION>
<OPTION value="3">Monday</OPTION>
<OPTION value="4">Tuesday</OPTION>
</SELECT>
<P>E-Mail Address:
<INPUT TYPE="text" NAME="EmailAddr">
<INPUT TYPE="submit" value="Register Me Please">
</FORM>
<SCRIPT LANGUAGE="JavaScript1.2">
<!--
function keypress_LastName (evt) {
evt.keyCode = cnvUCASE(evt.keyCode);
}
function cnvUCASE (K) {
if (K >= 97 && K <= 122) K = K - 32;
return K;
}
// -->
</SCRIPT>
</CENTER></BODY>
</HTML>
```

As you can see, the Input tag for the field “LastName” has been enhanced to call a JavaScript function named “keypress_LastName” when a key is pressed. From this function another function named “cnvUCASE” is called to examine the value of the keyboard character to see if it is in the range of 97-122, the ASCII codes for lowercase “a”- “z”. If any of those values are detected, they are modified by subtracting 32, which results in the ASCII uppercase “A” – “Z”. Field masking for Social Security numbers or telephone numbers is done much the same way.

I believe that it is counterproductive to include printouts of program coding for every function and feature. I will instead focus on the online forms system architecture, and the benefits of forms systems created with HTML and JavaScript.

Form Containers, Content, and Controls

An important concept to introduce in an intermediate HTML class is the idea of Form Containers, Content, and Controls. HTML forms exist primarily as *containers* for fill data. The fill data that is keyed or computed or retrieved and displayed inside fill fields is called *content*. The buttons, tabbing order, field highlighting, and other navigational functions within the form are called *controls*. The controls for an HTML form also include web server and database connectivity. To successfully implement and deploy Internet forms solutions, one must be equipped to create and maintain Form Containers, Content, and Controls.

Although the basic HTML tags allow online forms to be developed with a simple text editor, it is not productive to build and maintain an enterprise solution by hand-coding HTML. There are great forms design and development tools from several of the companies sponsoring the BFMA Symposium. Form development is simplified by using software that automatically converts a form design into HTML code while generating the server-side form handler software.

Creating Database-Connected HTML Forms

After designing an HTML form, there are several steps necessary to make database connections. For OFDP users, these steps are simplified through menu functions and dialog panels that help the form designer to:

- Insert a Submit button
- Insert necessary hidden fields
- Link data tables
    - Input tables (Read Only)
    - Output tables (Allow Write)
- Link form fields to data table columns
- Identify input Trigger fields

The first step is to make the I-form submittable by inserting a *Submit* button. Note that the OFDP Submit button uses the “HTML Post” method for transmitting the field name/value pair data to a server script URL.

Several hidden fields must also be inserted onto the I-form so that when the I-form is served to the client, navigational information and transaction status are preserved. This insures that the correct next page will be presented to the client when a form is submitted.

Before making database connections, it is important to identify the data tables that are going to be used with the I-form. Typically there is an *Output* table where form fill data is stored. Sometimes there are one or more *Input* tables used to populate fields on the I-form. Oftentimes, an Output table for an I-form is later used as an Input table for a subsequent I-form.

Next, each I-form fill field is linked to a Output data table column name. Some fields may have links to both Input and Output tables. In many cases, the I-form field list is used to create a new Output table where each I-form field name is mirrored as a table column name.

Finally, for each input table, there must be a “trigger” event to force a data table record to be retrieved and I-form fill fields to be populated. Typically a fill field is designated as a trigger field, and an event occurs when the client keys a value (i.e. account number) into the field and presses the tab key. This causes other fields (i.e. name and address) to be populated.

HTML Forms Interface to Web Server

There is one other significant technical issue to explain. Since fillable forms are most valuable when the field content can be captured on a web server, a complete HTML forms design solution must manage the web server interfacing. The basic web server functions allow forms to be opened, filled, submitted, and saved. These server functions are usually programmed using web server scripting languages such as CGI/Perl or ASP.

A *web server script* is a command list that is executed by an Internet web server to direct the page management processes. The *forms handler scripts* provide the critical link between submittable forms and the server database management system. Some of the most common web server scripts are listed below:

- Web Server Scripts
  - *Forms Handler Scripts*
    - Open new unfilled form
    - Open filled form
    - Open flattened form
    - Query and retrieve data from tables
    - Save submitted form data

For pre-populated forms, the *open filled form* script retrieves the appropriate data record and fills the necessary form fields before serving the form to the client. For submittable forms, the *save submitted form data* script stores the field data into a table record. For dynamic database view forms, the *query and retrieve field data* sends a query when the client keys in a data value (i.e. account number) and returns data values to repopulate multiple form fields. This script can also retrieve a collection of values to populate a drop-down list. For field-flattened forms, the *open flattened form* script retrieves the appropriate data record and replaces fill fields with inline text before serving the form to the client.

Another class of scripts assists users and administrators in organizing their forms and building forms-oriented workflows:

  - *Client Support Scripts*
    - Search for records
    - List records
    - E-mail records

  - *Administrator Support Scripts*
    - Create/Drop data tables
    - Examine data tables
    - Delete records
    - Import/Export record data as XML

- User Access Control Scripts
  - Manage Login Password/ID
  - User Profiles
  - Administrative (Who has access to Which forms)

- Workflow Processing Scripts
  - Approvals
  - Tracking
  - Reporting
  - Connectivity to other Business Systems

The Components of an Online I-forms System

An online I-forms system has essentially the same architecture as an e-commerce system, without the shopping cart and payment processing modules. Instead, the I-forms system may include expanded workflows for digital signatures, form approvals, and tracking.

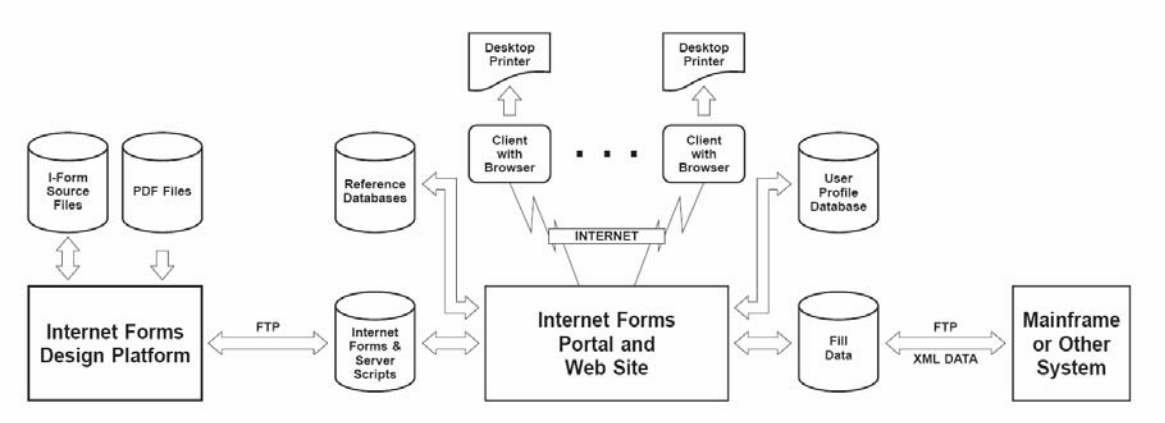


Diagram of a Typical Online Internet Forms System

The above architecture is ideal for centralized forms and database management on a large scale, using the Internet as the communication medium. This method is typically handled by utilizing the following technologies:

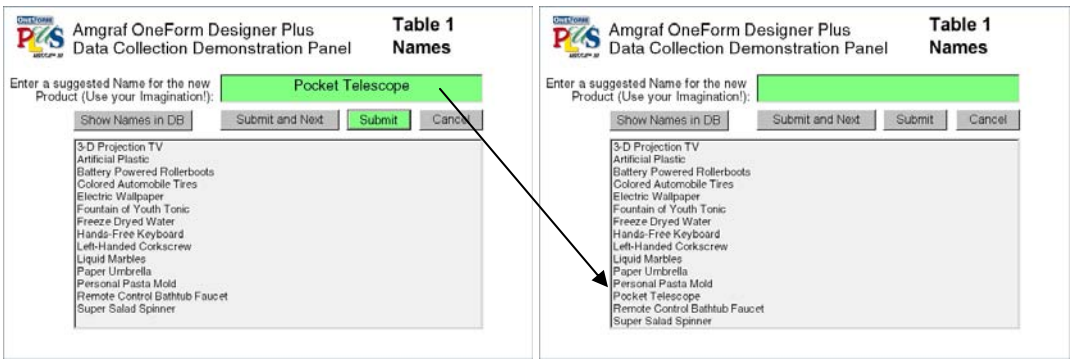
- Server-side Relational Database Management System (RDBMS)
- Web Server to Collect and Retrieve Form Field Data
- Internet Forms Repository
- Windows PC's with the Microsoft Internet Explorer (or compatible) Browser

The required software components are:

- Relational Database Management System (RDBMS)
  - Oracle
  - SQL Server 2000
  - MySQL
  - DB2
- Forms Repository
  - Save by Classification and Category
  - Save by Data Format (HTML, PDF, Word, etc.)
  - Maintain Form Versions
- Web Server
  - IIS
  - Apache

Demonstration of HTML Forms Database Connectivity

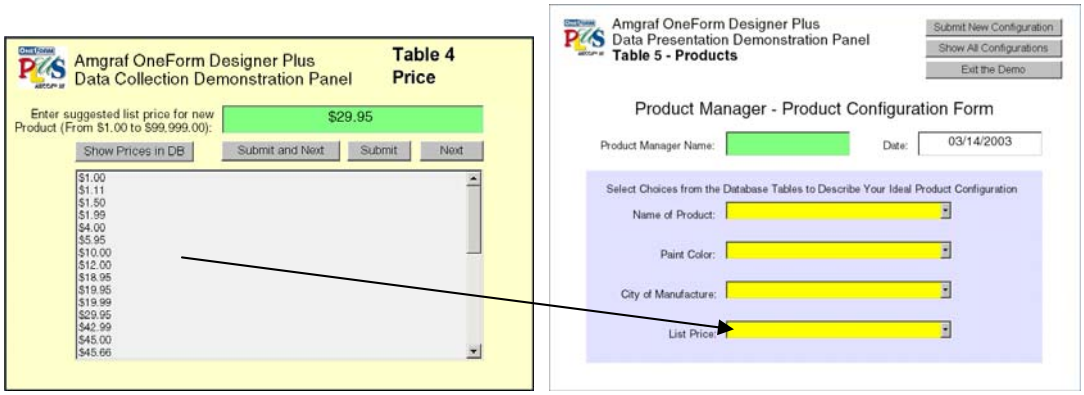
There are several I-form demonstrations located on Amgraf's web site at www.amgraf.com.
One of the examples is illustrated below. In this simple demonstration, there are five consecutive HTML I-forms used to capture and configure product information. Each I-form saves the information into a separate Output table. The tables are called "Names", "Colors", "Cities", "Price", and "Products".



A list box on the I-form instantly updates the contents as new information is submitted.

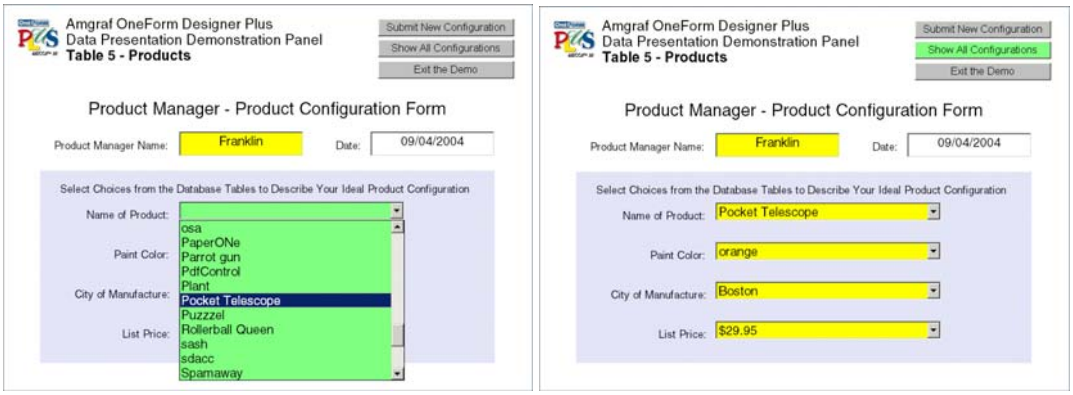
The "Show Names in DB" button displays a current list of the data values in the Output table. New fill data is stored into the Output table when the user clicks the "Submit" button.

I-forms to Capture a Color and City Location Name.



I-form to Capture Product Prices that will Populate Dropdown List on the Products I-form.

The Output tables from the first four I-forms are now used as Input tables for the Products I-form. The dropdown list fields are populated with data captured in the first four I-forms.



The Dropdown List Fields on the Configuration I-form contains data from the Input Tables.

The user can select from the drop-down lists and then click the "Submit Configuration" to save the choices. Clicking the "Show All Configurations" button produces a report that shows the 20 most recent user choices.

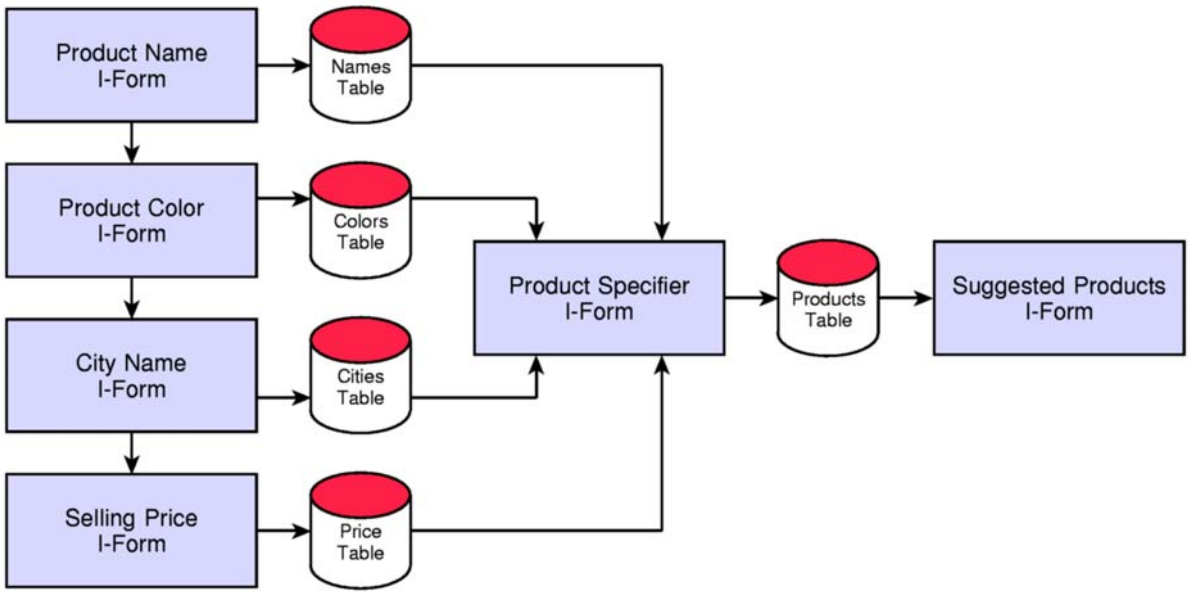
Amgraf OneForm Designer Plus
Database Presentation and Reporting

Close Window

The 20 Most Recent Product Configurations

Manager	Product	Color	City	Price	Date
Franklin	Pocket Telescope	orange	Boston	$29.95	9/4/2004
Richard Noggin	Liquid Ice	Chartruse	Wolfspit Kansas	$309.77	9/1/2004
Joe Schmoe	Goat Glue	Jerked Chicken Brown	DuckSquat Texas	$5.95	8/31/2004
Steve	Dr. Zogg's Secret Formula Coug	Green	sacramento	$700.00	8/30/2004
Robert	PaperONe	Blue	Jakarta	$19.95	8/26/2004
Diana	Brand-New-Product	perrywinkle	sacramento	$700.00	8/24/2004
Joe	BOSCAD	Bilious Blue	jamestown	$10.00	8/20/2004
blah	a	blue	baltimore	$18.95	8/18/2004
Podna	Puzzzel	pizza	paloma	$123.00	8/15/2004
Eric E	NEO	Blue	Gloucester	$10,000.00	8/9/2004
Joe Smoe	new product 4	purple people	Taipai	$1.11	7/23/2004
Fred	Brand-New-Product	65464645	baltimore	$29.95	7/16/2004
tim	Goat Glue	red	sacto	$19.95	7/12/2004
Dennis	tester321	purple	Ottawa	$99.99	7/6/2004
svn	Spamaway	Red and Blue	Adipur	$999.00	6/26/2004
gILLES	Aculert	65464645	Boston	$5.95	6/14/2004
Kahuna	Open Ocean Surfboards	Red and Blue	Sidney	$2,000.00	6/1/2004
asdas	Brand-New-Product	Magenta	New Orleans	$5,000.00	5/27/2004
Romulus	Spamaway	Bilious Blue	mississauga	$29.95	5/24/2004
Romulus	Rollerball Queen	Jerked Chicken Brown	DuckSquat Texas	$29.95	5/24/2004

Product Configuration Report that contains Data Captured with I-forms



I-forms Database Connectivity Diagram for the Online Demonstration.

Using HTML and PDF Internet Forms Interchangeably

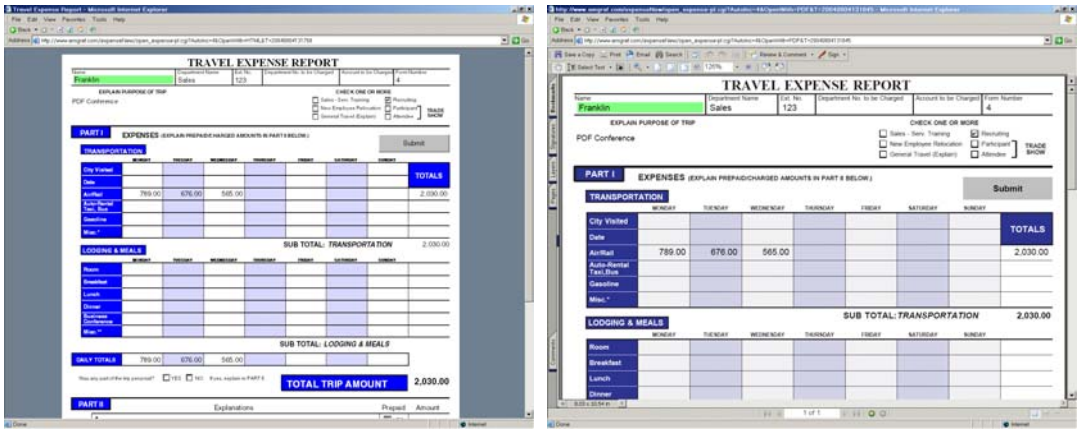
Amgraf's OFDP software can also produce PDF forms with database connectivity as described in this presentation. OFDP can also generate HTML versions of PDF forms that look and act the same without the need to start the Adobe Reader. Amgraf's server scripts work equally well with both HTML and PDF forms. From our experience in helping to implement large-scale I-forms solutions, we have found that there are often advantages to using HTML forms along with PDF forms. We believe that HTML forms are best suited for:

- Online data collection
- Dynamic database views
- Use with external JavaScript functions

We recommend PDF forms for:

- Printing
- Field-flattened I-forms
- E-mail attachments
- Archival purposes

As the screen shots shown below illustrate, there are many similarities between the HTML and PDF I-forms produced by Amgraf's OFDP software.



This Database-Connected HTML I-form (Left) looks like a PDF I-form.

You can try out the example forms shown here by going to Amgraf's web site demonstration page at www.amgraf.com/pages/iforms.html.

Conclusions about HTML Forms

HTML forms are the most popular forms methodology for data capture and data presentation on interactive web sites. HTML forms can be generated to look and fill like standard paper forms, or they can be developed as simple caption-blank input screens. Field validation, database connections, and e-mail links are easily established during the forms design process.

Paper business forms, especially internal forms, are prime candidates for replacement with HTML forms. The forms can be hosted on Internet/intranet systems, fill data can be captured, and information managed more efficiently. We believe there is great value in preserving the look and feel of the paper form as it is made electronic because end-users recognize the form, understand how to fill it out, and the form can still be printed when needed.

Although the basic HTML tags allow online forms to be developed with a simple text editor, it is not productive to build and maintain an enterprise solution by hand-coding HTML. There are great forms design and development tools from several of the companies sponsoring BFMA events like this Forms Institute. Form development is simplified by using software that automatically converts a form design into HTML code while generating the server-side form handler software. Using these tools, many more functions and features can be implemented without needing to know the nuances of HTML. Input masking (i.e. inserting dashes into a phone number), keystroke validation, database connectivity, dynamic paging, and entire workflows can be streamlined using commercial software tools to generate the HTML along with JavaScript and other programming languages.

About Amgraf, Inc.

Amgraf is the developer of the OneForm Designer Plus and OneForm Manager Lite software technologies for designing and deploying "paper-friendly" e-business forms. E-business forms provide the structure for user interaction with business processes, including eCommerce and eGovernment.

We are the clear market leader in forms design. We provide a robust forms development platform, with an overall philosophy of "One design session – many uses". *Most importantly, we never require filler software or licenses to use e-forms created with our tools.* And, we provide total support – "We're with you all the way" – including training, design and mapping services, consulting and technical services. For more information please go to our web site at http://www.amgraf.com/.

Bibliography

Some of the content of this presentation was derived from the following sources:

HTML 4.01 Reference
http://www.w3schools.com/html/html_reference.asp

RichInStyle.com HTML 4 guide – Forms
http://www.richinstyle.com/guides/forms4.html

How to write HTML forms
http://www.cs.tut.fi/~jkorpela/forms

HTML Form Tags Examples
http://www.mountaindragon.com/html/forms.htm

EditPlus Text Editor
http://www.editplus.com/