Utility Data Web Page Design—Learning Technologies

David C. Green
Green Management Services, Inc.
Fort Myers, Florida

Fangxing Li
ABB Inc.
Raleigh, North Carolina

ABSTRACT

Creating web pages requires the use of at least a few of the several technologies available to developers today. Reliable, robust, informative and intuitive web pages displaying utility data can be created using hypertext markup language (HTML) and common gateway interface (CGI). Other technologies enhance the presentation and functionality of web pages. We discuss these technologies from the viewpoint that a great deal of application development today is undertaken through successive revisions. We present technology descriptions, along with a simple example of each, to get a developer started or simply for better understanding. This is not an exhaustive guide to using each of the technologies, but rather an "opening the door" approach. We then provide references and links to resources for further study.

INTRODUCTION

In order to web-enable utility data, the user interface (UI) must be embedded in a web browser like Microsoft Internet Explorer (IE) or Netscape Navigator. Existing and recently released technologies to implement this are:

- Plain HTML;
- Server-side scripts such as common gateway interface, Active Server Pages, Java Server Pages, Java Servlets, and PHP;
- Client-side scripts using JavaScript, and VBScript;
- Java applets;
- ActiveX controls.

HTML and dynamically generated web pages are usually employed to create a UI embedded in a web browser. In general, HTML is only able to display data and submit simple requests. Server- or client-side scripts can be a good choice for non-static web pages requiring more user interaction, database support, and/or constant updates.

ACCESSING THE DATA SOURCE

Applications, whether running on a desktop or over the Internet, often need to read and write information to and from data storage. A desktop database like Microsoft Access or flat files may be suitable for traditional single-user applications or even client-side web applications with relatively straightforward data requirements. A distributed database, such as Oracle or Microsoft SQL Server, is more suitable for enterprise applications requiring better performance, better scalability, and more features such as multiple controlled access, transaction roll back, and remote access and administration.

There are many different database drivers that provide access to databases. Microsoft has provided several technologies for MS Windows applications. Data access objects (DAO), open database connectivity (ODBC), and remote data objects (RDO) are earlier technologies which are still utilized for many existing applications. Microsoft has recently replaced those technologies with a single model, universal data access (UDA), which provides access to a variety of relational and non-relational information sources. UDA consists mainly of the high-level interface called ActiveX data objects (ADO) and the lower-level services called OLE DB.

Sun Microsystems offers Java Database Connectivity (JDBC), an application programming interface (API) that can be used to access virtually any tabular data source from the Java programming language. JDBC provides cross-DBMS connectivity to a wide range of SQL-based

databases, and also provides access to other tabular data sources, such as spreadsheets.

XML (extensible markup language) is a tag-driven technique to create structured information and share both the structure and the data on the internet, intranets, and elsewhere. Since its standard format is plain text, it may be inefficient when applied to engineering applications involving mass data processing.

A simple, but efficient way of storing and accessing data is the use of binary format data files. The main advantage is that they are generally small and efficient, so they can be transported very fast over the internet. There are also no additional requirements for database drivers. Object serialization techniques can be utilized to automatically create binary files. The main disadvantage in this context is that they are not human-readable. When the data are not explicitly exposed to and manipulated by users, this may be a viable solution.

PUTTING IT ALL TOGETHER

The developer may have many different options for implementing web-enabled technologies. Some of the factors that might influence a developer's design decision are:

- User requirements;
- Platform constraints;
- Migration from the legacy system;
- Time to market;
- Development cost;
- Difficulty of integration.

For a technology to ultimately work, data must be passed efficiently and seamlessly between the various components. Communication between the components can range from basic scripting and simple text file transfers to more complex web-service techniques such as SOAP (simple object access protocol) and CORBA (common object request broker architecture). However, generally speaking, the more complex the communication protocol, the more costly it may be to implement and maintain. (1)

The developer can choose options from a wide range of technolo-

gies discussed in the remainder of this chapter. Some have advantages over others as to reliability, complexity, and cost. Some technologies may be more on the cutting edge, but less reliable and more costly. The difficult decision the developer has to make is whether to use the latest and greatest or stick with well-proven technologies. We present some of these technologies to you in an objective form with little or no personal bias. It depends on the particular application as to which technologies will work best.

HTML

***H**yper**T**ext **M**arkup **L**anguage* (HTML) is the authoring language used to create documents on the World Wide Web. HTML defines the structure and layout of a web document by using a variety of tags and attributes. Tags are also used to specify hypertext links. These allow web developers to direct users to other web pages with only a click of the mouse on either an image or word(s). HTML is a non-proprietary format, and can be created and processed by a wide range of tools, from simple plain text editors to sophisticated WYSIWYG authoring tools.

Tags

A tag is a series of characters, with no spaces, placed between the angle brackets < and >. It is usual to write the commands in upper case like **<TITLE>**, **<BODY>** in textbooks to show the tags more clearly, but in practice it is best to keep everything in HTML lower case. Most of the HTML editors produce lower case code.

Some tags are used **singly**; for example, those that stop display on one line and have it continue on the next:
. These are usually called *markup commands*. Most tags are used in **pairs**. The end tag is identical to the start tag except for one difference: its name is preceded by a slash. These are called container tags. The command has an effect on the text placed between the start and end tags.

Example: Utility data for **<B>**4/13/2003**</B>** will be displayed as follows: Utility data for **4/13/2003**

The tag <B> indicates that the text be displayed in bold. It is possible to add tags together to display more than one change.

Example: Utility data for **<B><I>**4/13/2003</I>**</B>** will be displayed as bold and italic text: Utility data for ***4/13/2003***
The tag <I> indicates italics.

Basic Layout of Tags to Start Your Web Page

Your web page is divided into two distinct sections, the **HEAD** and the **BODY**. The entire file is enclosed in a <**HTML**> tag. The head is placed in a <**HEAD**> container tag. This section often has only one other command, <**TITLE**>, which specifies a general title for the site. The title will be displayed in the **title bar** of the browser. No other part of the head is displayed on the screen. The rest of the file is contained in a <BODY> container tag. This section will contain the various elements that make up the web page (text, images, tables, line breaks etc.).

A skeleton page would be in the following order:

<HTML>
<HEAD>
<TITLE> Put your title here**</TITLE>**
</HEAD>
<BODY>

Everything else goes here. This is the visible part of the page.
</BODY>
</HTML>

Adding Links to Other Pages

What makes the Web so effective is the ability to define links from one page to another, and to follow links at the click of a button. Links are defined with the <a> tag. Let's define a link to the page in the file "utilitydata.html." This is a link to Today's Utility Data. The text between the <a> and the is used as the caption for the link. It is common for the caption to be in blue underlined text. To link to a page on another website, you need to give the full web address (commonly called a URL). For instance, to link to www.utilityreporting.com you need to write:

This is a link to
UtilityReporting.Com.
You can turn an image into a hypertext link; for example, the following allows you to click on the company logo to get to the home page:

<img src="logo.gif" alt="Home Page">

Setting Colors

The color of a page background and default colors for text can be set in the <**BODY**> tag. The color of selected blocks of text can be set in the <**FONT**> tag. There are two ways to define colors. You can name them using a color name or give them a hexadecimal value. Unfortunately, only 16 color names can be recognized. However, hexadecimal can pick from 24 million colors. It is worth remembering, however, that many people have only a 256-color monitor.

In summary, knowing all about HTML straightaway is not essential. You will learn the basics as you build pages. You will eventually recognize the tags and be able to read HTML just as easily as reading a chapter in a book. Then you can begin to modify the code to suit your pages or make things happen that your software will not allow you to do. The best thing you can do is go to a discount book shop and buy the biggest book on HTML version 4 you can find at the lowest price. There are plenty of such books around and they are now out of print (or are several years old) and considered by the big book shops to be un-sellable. There are good resources on the web to explain the basics of HTML. Reference sites such as The Web Designer's Virtual Library [http://www.wdvl.com] allow you to look up the HTML tags.

XHTML

XHTML is an acronym for "eXtensible HyperText Markup Language," a reformulation of HTML 4.0 as an XML 1.0 application. XHTML provides the framework for future extensions of HTML and aims to replace HTML in the future. (2)

XHTML 1.0 is the first step toward a modular and extensible web based on XML (extensible markup language). It provides the bridge for web designers to enter the Web of the future, while still being able to maintain compatibility with today's HTML 4.0 browsers. It looks very much like HTML 4.0, with a few notable exceptions, so if you're familiar with HTML 4.0, XHTML will be easy to learn and use. (3)

XHTML in a nutshell

- XHTML tags are all lowercase.

- XHTML is a stricter, tidier version of HTML.
- Pages written in XHTML work well in most browsers.
- All tags, including empty elements, must be closed.
- XHTML is the reformulation of HTML 4.0 as an application of XML.
- The elements (tags) and attributes are almost identical to HTML.

View an example at: [http://utilityreporting.com/xhtml/example1.htm]. (4)

XHTML 1.0 became an official W3C Recommendation January 26, 2000. A W3C Recommendation means that the specification is stable, that it has been reviewed by the W3C membership, and that the specification is now a web standard. (5)

The W3C MarkUp Validation Service [http://validator.w3.org/] is a free service that checks documents like HTML and XHTML for conformance to W3C Recommendations and other standards. (6)

CASCADING STYLE SHEETS (CSS)

When cascading style sheets were introduced in late 1996, they represented an exciting new opportunity. They enabled much more sophisticated page design (typography and layout) than web developers had been used to, and they helped manage the complex tasks of developing and maintaining sites and keeping them up to date.

Why use CSS?

With CSS, you can decide how headings should appear, and enter that information once. Every heading in every page that is linked to this style sheet now has that appearance. If you want to make every heading of level 3 more obviously different from those of level 2, just edit the style sheet and every such heading now has the altered appearance.

What Exactly is a Style Sheet?

A style sheet is simply a **text file** (which has a .css suffix) written according to the grammar defined in the CSS1 or CSS2 recommendations of the W3C.

Here is a simple example.

```
body
{font-family: Verdana, "Minion Web," Helvetica, sans-serif;
font-size: 1em;
text-align: justify;
background:#00ff00;
color:#ffffff}
h1
{font-family: Verdana, sans-serif;
font-size: 1.3em}
```

In addition to being in .css files, style sheets can also be **embedded** into the *head* element of HTML files.

Embedding Style Sheets

Style sheets can be embedded into the *head* element of HTML documents. Quite simply, the style sheet itself is placed between a style tag like this:

<style type="text/css"> </style>

Embedding means that you lose one of the major advantages of CSS, which is the ability to modify the appearance of a whole site by making changes to a single file.

Linking to Style Sheets

Note that the web page links to the style sheet. The style sheet has no knowledge of the pages which are linked to it. To link a web page to a style sheet, you place a link to the style sheet in the **head** of the document, using the following syntax:

<link rel="stylesheet" type="text/css" href="http://utilityreporting.com/styles/style1.css" /> (7)

View the results of the example at: [http://utilityreporting.com/css/example1.htm]

The W3C CSS Validation Service [http://jigsaw.w3.org/css-validator/check/referer] is a free service that checks documents like Cascading Style Sheets for conformance to W3C Recommendations and other standards.

DYNAMIC HTML

DHTML is not a singular technology but a combination of three existing technologies glued together by the document object model (DOM).

HTML - For creating text and image links and other page elements.
CSS - Style sheets for further formatting of text and HTML, plus other added features such as positioning and layering content.
JavaScript - The programming language that allows you to access and dynamically control the individual properties of both HTML and style sheets.

The way JavaScript accesses the properties of an HTML document is through the document object model (DOM). The job of the DOM is to expose all the attributes of HTML and style sheets to JavaScript control. All you need to know about the DOM is what JavaScript commands it will accept. Not so easy, as different browsers have their slightly different versions of the DOM, so they access HTML properties differently as well as display them differently.

An excellent DHTML reference book is *Dynamic HTML—The Definitive Guide* by Danny Goodman (O'Riley Books). It lists all of the DHMTL properties and their cross browser compatibilities. (8)

THE DOCUMENT OBJECT MODEL (DOM)

We'll start with the basics, a very simple HTML page.

```
<html><head></head><body bgcolor="white"><div id="a" style="font-
family:
Arial; color: white; background: black">Document Objects</div></body></
html>
```

Under the new DOM, every element in an HTML document is part of a "tree," and you can access each and every element by navigating through the tree "branches" until you reach the corresponding "node."

```
document
    | — <html>
        | — <head>
        | — <body>
            | — <div>
```

The DOM offers a fast and efficient method of accessing elements within the page using the getElementById() method.

Let's alter the font color of the text within the .

```
<script language="JavaScript">
var obj = document.getElementById("a");
obj.style.color = "red";
</script>
```

Now that you know how to find your way to specific HTML elements in the document, it's time to learn how to manipulate them. (9)

Consider the following example at: [http://utilityreporting.com/dhtml/example1.htm].

Select "View Source" in your browser to see how the DOM is used.

XML

An XML document is a database only in the strictest sense of the term. That is, it is a collection of data. As a database format, XML has some advantages. For example, it is self-describing (the markup describes the structure and type names of the data, although not the semantics), it is portable, and it can describe data in tree or graph structures. It also has some disadvantages. For example, it is verbose, and access to the data is slow due to parsing and text conversion.

It may be possible to use an XML document or documents as a database in environments with small amounts of data, few users, and modest performance requirements. However, it will fail in most production environments, which have many users, strict data integrity requirements, and the need for good performance.

Data-Centric Documents

Data-centric documents are documents that use XML as a data

transport. They are designed for machine consumption, and the fact that XML is used at all is usually superfluous. That is, it is not important to the application or the database that the data are, for some length of time, stored in an XML document. Data-centric documents are characterized by regular structure, fine-grained data, and little or no mixed content.

Storing and Retrieving Data

To transfer data between XML documents and a database, it is necessary to map the XML document schema to the database schema. The data transfer software is then built on top of this mapping. The software may use an XML query language (such as XPath, XQuery, or a proprietary language) or simply transfer data according to the mapping (the XML equivalent of SELECT * FROM Table). (10)

SERVER-SIDE APPLICATIONS

The first generation of server-side technologies is common gateway interface (CGI), usually developed in Perl programming language. This technology employs server-side applications to complete a predefined task when a client's request is received. This may be far from efficient since the server-side application is heavy-weighted and consumes many server resources. The next generation of server-side technologies is server-side scripts such as active server pages, Java Servlet/Java server pages, PHP, etc. These technologies allow the server to generate a thread to handle a client's request. The thread is light-weighted and certainly consumes much fewer resources than the application generated by CGI. Hence, this is more efficient than the CGI technology. (1)

Common Gateway Interface (CGI)

The common gateway interface (CGI) is a standard for interfacing external applications with information servers, such as HTTP or web servers. A plain HTML document that the web server **retrieves** is **static**, which means it exists in a constant state: a text file that doesn't change. A CGI program, on the other hand, is **executed** in real time, so that it can output **dynamic** information.

For example, let's say that you wanted to hook up your Unix database to the World Wide Web, to allow people from all over the world to query it. You need to create a CGI program that the web server will

execute to transmit information to the database engine, and receive the results back again and display them to the client. This is an example of a *gateway*, and this is where CGI got its origins. The one thing you need to remember is that your CGI program should not take too long to process. Otherwise, users will just be staring at their browsers waiting for something to happen. (11)

The basics of a CGI script do not require much effort to create. A simple program to print the current date requires only a few lines of PERL:

Example script #2

```
#!/usr/local/bin/perl5
# # This tells the client to expect an HTML document
#
print "Content-type: text/html,""\n\n";
$today='date'; # Get the date
chop $today;
# Remove the newline on the end
print "Today's date is $today. \n";
$this_server=$ENV{'SERVER_NAME'};
print "You can <A HREF=\"http://$this_server/\">return to the server</
A>";
exit(0);
```

See what it looks like in the browser at: [http://utilityreporting.com/cgi-bin/cgi-example2.pl]

There are numerous resources for information on the common gateway interfaces available on the Web. Some of the more valuable include: (12)

- NCSA [http://hoohoo.ncsa.uiuc.edu/cgi/interface.html] has the canonical copy of the CGI specifications as well as a simplified primer [http://hoohoo.ncsa.uiuc.edu/cgi/primer.html].
- Thomas Boutell's *Web FAQ* [http://www.boutell.com/faq]
- O'Reilly produces a book, *CGI Programming on the World Wide Web* [http://www.oreilly.com/catalog/cgi/].

Active Server Pages (ASP)

Files created with active server pages have the extension .ASP.

With ASP files, you can activate your website using any combination of HTML, scripting (JavaScript or VBScript) and components written in any language. When you make a change on the ASP file on the server, you need only save the changes to the file and the next time the web page is loaded, the script will automatically be compiled. It works because ASP technology is built directly into Microsoft web servers, and is thus supported on all Microsoft web servers.

ASP uses the delimiters "<%" and "%>" to enclose script commands. The scripting languages supported by ASP in turn support use of the If-Then-Else. Finally, you can embed some real logic into your HTML. For example, the following code from the IIS documentation shows how you can set the greeting shown based upon the time of day. <FONT COLOR="GREEN"> <%If Time >= #12:00:00 AM# And Time < #12:00:00 PM# Then%> Good Morning! <%Else%> Hello! <%End If%> </FONT>

Here is a simple ASP page.

Source file:

<html> <head><title>My first ASP page</title> </head> <body> The current date is <%=Date%>. </body> </html>

This is how it looks in a web browser:

The current date is 3/24/2003.

Try this URL: [http://www26.brinkster.com/dcgreen/test.asp] to view the results.

JavaTM Servlet

JavaTM Servlet technology provides web developers with a simple, consistent mechanism for extending the functionality of a web server and for accessing existing business systems. A servlet can almost be thought of as an applet that runs on the server side. Servlets provide a component-based, platform-independent method for building web-based applications, without the performance limitations of CGI programs. In addition, unlike proprietary server extension mechanisms

(such as the Netscape Server API or Apache modules), servlets are server and platform independent. Today, servlets are a popular choice for building interactive web applications. Third-party servlet containers are available for Apache Web Server, Microsoft IIS, and others. Servlet containers are usually a component of web and application servers, such as BEA WebLogic Application Server, IBM WebSphere, Sun ONE Web Server, Sun ONE Application Server, and others. (13)

JavaServer Pages

JavaServer Pages (JSP) technology is an extension of the JavaTM Servlet technology. Together, JSP technology and servlets provide an attractive alternative to other types of dynamic web scripting and programming. It offers platform independence, enhanced performance, separation of logic from display, ease of administration, extensibility into the enterprise and, most importantly, ease of use.

Web developers and designers use JavaServer Pages technology to rapidly develop and easily maintain information-rich, dynamic web pages that leverage existing business systems. JavaServer Pages technology uses XML-like tags that encapsulate the logic that generates the content for the page. Additionally, the application logic can reside in server-based resources (such as JavaBeansTM component architecture) that the page accesses with these tags. All formatting (HTML or XML) tags are passed directly back to the response page. By separating the page logic from its design and display and supporting a reusable component-based design, JSP technology makes it faster and easier than ever to build web-based applications. (14)

PHP

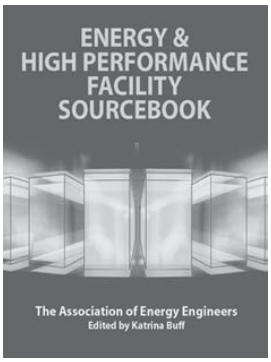
PHP is a language for easily building dynamic web pages. It is ideally suited to the web because PHP scripts live inside web pages right along with the HTML tags and content. For that reason, PHP is called an embedded scripting language. However, unlike some web scripting languages, PHP makes a clear distinction between sections of PHP code and sections of the HTML document. When the web server fills a request for a PHP enabled page, it first looks through the page content for sections of PHP code and executes any it finds. Any normal HTML sections are passed to the browser without any changes.

Here are a few good reasons to choose PHP for enabling interactive content on your web site:

Learn from the nation's leading authorities…

ENERGY & HIGH PERFORMANCE FACILITY SOURCEBOOK

NEW



The Association of Energy Engineers
Edited by Katrina Buff

This comprehensive volume brings you in a single volume a comprehensive resource covering the very latest technological developments for sustainable buildings. You'll benefit from the expertise contributed by more than 90 noted authorities, with detailed coverage of subjects including "green building" design and operation, building commissioning, methodologies for improving both the energy performance and indoor quality of buildings, and much more. You'll find reports on the very latest developments in daylighting, combined heat and power, distributed generation, HVAC optimization, power quality, efficient lighting, solar thermal energy, solar heating and air conditioning, energy load profiling and management, project financing opportunities, and more, including a wealth of case studies.

ISBN: 0-88173-420-9

ORDER CODE: 0525

8½ x 11, 670 pp., Illus.
Softcover, $95.00

CONTENTS

BOOK ORDER FORM

① Complete quantity and amount due for each book you wish to order:

Quantity	Book Title	Order Code	Price	Amount Due
	Energy & High Performance Facility Sourcebook	**0525**	**$95.00**	

② Indicate shipping address: **CODE: Journal 2003**

NAME (Please print) BUSINESS PHONE

SIGNATURE (Required to process order)

COMPANY

STREET ADDRESS ONLY (No P.O. Box)

CITY, STATE, ZIP

Applicable Discount	
Georgia Residents add 6% Sales Tax	
Shipping Fees	**9.00**
TOTAL	

MEMBER DISCOUNTS
A 15% discount is allowed to AEE members.
☐ AEE Member (Member No._____)

③ Select method of payment:
☐ **CHECK ENCLOSED**
☐ **CHARGE TO MY CREDIT CARD**
☐ *VISA* ☐ *MASTERCARD* ☐ *AMERICAN EXPRESS*

Make check payable in U.S. funds to:
AEE ENERGY BOOKS

CARD NO.

Expiration date *Signature*

√ Send your order to:
AEE BOOKS
P.O. Box 1026
Lilburn, GA 30048

INTERNET ORDERING
www.aeecenter.org

TO ORDER BY PHONE
Use your credit card and call :
(770) 925-9558

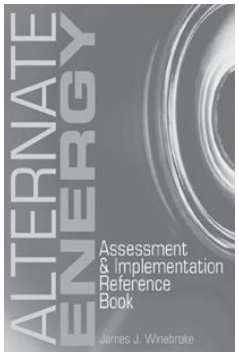
TO ORDER BY FAX
Complete and Fax to:
(770) 381-9865

INTERNATIONAL ORDERS
Must be prepaid in U.S. dollars and must include an additional charge of $10.00 per book plus 15% for shipping and handling by surface mail.

Get up to speed on the complete spectrum of future energy technologies and their true market potential…

ALTERNATE ENERGY: ASSESSMENT & IMPLEMENTATION REFERENCE BOOK

NEW



By James J. Winebrake, Ph.D.

Here's your opportunity to look into the future of energy technologies, with emphasis on alternative, or non-conventional technologies, their potential impacts, and the technical, economic and policy issues that will affect their successful integration into global energy markets. Over the past several years, industry and government have turned to a strategic planning technique called "roadmapping" to help assess future energy management practices and technologies. This book considers energy management and technology development over the next several decades by exploring data from these energy technology roadmaps. International in scope, the book examines both the technical and non-technical aspects of emerging technologies. Detailed technology assessments for specific alternative energy resources are presented. An overview of the problems associated with conventional energy consumption is included, as well as an insightful discussion of technology implementation issues from the author's own well-informed and cautiously optimistic perspective.

ISBN: 0-88173-436-5

ORDER CODE: 0515

6 x 9, 243 pp., Illus.
Hardcover, $125.00

CONTENTS

BOOK ORDER FORM

① Complete quantity and amount due for each book you wish to order:

Quantity	Book Title	Order Code	Price	Amount Due
	Alternate Energy: Assessment & Implementation Reference Book	**0515**	**$125.00**	

② Indicate shipping address: **CODE: Journal 2003**

NAME (Please print) BUSINESS PHONE

SIGNATURE (Required to process order)

COMPANY

STREET ADDRESS ONLY (No P.O. Box)

CITY, STATE, ZIP

Applicable Discount	
Georgia Residents add 6% Sales Tax	
Shipping Fees	**9.00**
TOTAL	

③ Select method of payment:

☐ **CHECK ENCLOSED**
☐ **CHARGE TO MY CREDIT CARD**
☐ *VISA* ☐ *MASTERCARD* ☐ *AMERICAN EXPRESS*

Make check payable in U.S. funds to:
AEE ENERGY BOOKS

CARD NO.

Expiration date *Signature*

Send your order to:
AEE BOOKS
P.O. Box 1026
Lilburn, GA 30048

INTERNET ORDERING
www.aeecenter.org

TO ORDER BY PHONE
Use your credit card and call :
(770) 925-9558

TO ORDER BY FAX
Complete and Fax to:
(770) 381-9865

INTERNATIONAL ORDERS
Must be prepaid in U.S. dollars and must include an additional charge of $10.00 per book plus 15% for shipping and handling by surface mail.

- It is open source.
- It uses similar syntax and constructs, knowledge of PHP can help you in learning the C language.
- The data types and structures of PHP are easy to use and understand, PHP knows what you mean and can convert types automatically.
- You don't have to know any special commands to compile your program, it runs right in your web browser.
- You don't have to know everything there is to know about PHP to start writing useful programs.
- PHP serves as a "wrapper" for many standard C libraries, which are easily compiled into the language giving it the flexibility to respond more rapidly to changes in web technology or trends.
- PHP enabled web sites can be deployed with amazing rapidity, due to its being tuned for dynamic pages and database backends.

You don't have to know everything there is to know about PHP before you can write useful programs. Here is a simple example. (15)

PHP 4 database example:

```
<?php
$dbname="php.dbf";
if (!$fp = dbase_open($dbname,0)) {
echo "Cannot open $dbname\n";
exit;
}
$nr = dbase_numrecords($fp);//Number of records.
echo "<Br>";
echo "Data Table<Br><Br>";
for ($i=1; $i <= $nr; $i++) {//From 1 to $nr as you know.
$temp = dbase_get_record($fp,$i);
//if (chop($temp[0]) == $frmName) {//$frmName comes from FORM via
WEB
echo "$temp[0], $temp[1], $temp[2] <Br>";
$data_array["column1"][$i-1] = $temp[0];
$data_array["column2"][$i-1] = $temp[1];
$data_array["column3"][$i-1] = $temp[2];
//}
}
array_multisort($data_array["column1"], SORT_STRING, SORT_ASC,
$data_array["column2"],$data_array["column3"]);
```

```
echo “<Br>”;
echo “Sorted by Column # 1<Br><Br>”;
echo $data_array[“column1”][0];
echo ,” “;
echo $data_array[“column2”][0];
echo ,” “;
echo $data_array[“column3”][0];
echo “<Br>”;
echo $data_array[“column1”][1];
echo ,” “;
echo $data_array[“column2”][1];
echo ,” “;
echo $data_array[“column3”][1];
echo “<Br>”;
echo $data_array[“column1”][2];
echo ,” “;
echo $data_array[“column2”][2];
echo ,” “;
echo $data_array[“column3”][2];
echo “<Br>”;
echo $data_array[“column1”][3];
echo ,” “;
echo $data_array[“column2”][3];
echo ,” “;
echo $data_array[“column3”][3];
dbase_close($fp);
exit; ?>
```

You can view the results of the example above at: [http://www.utilityreporting.com/php/dbase.php4].(1)

CLIENT-SIDE TECHNOLOGIES

There are a number of popular client-side technologies available to implement extra functionality on top of HTML. This section discusses three standard client-side technologies: client-side scripts, Java applets, and ActiveX controls.

Client-side scripts are probably the simplest client-side technology. A client-side script is just a collection of commands interpreted, rather than compiled, by the web-browser. Client-side scripts are relatively easy to implement, but limited in functionality compared with full-featured programming languages. Another drawback is that the code can

be viewed from the browser. Hence, any proprietary knowledge should not be implemented in client-side scripts. The most popular client-side script language is JavaScript. However, it is not the only one. VBScript can also be used for client-side script programming.

JavaScript

JavaScript is used in millions of web pages to improve the design, validate forms, and much more. JavaScript was developed by Netscape and is the most popular scripting language on the internet. JavaScript works in all major browsers that are version 3.0 or higher.

- JavaScript was designed to add interactivity to HTML pages.
- JavaScript is a scripting language—a scripting language is a light-weight programming language.
- A JavaScript is lines of executable computer code.
- A JavaScript is usually embedded directly in HTML pages.
- JavaScript is an interpreted language (means that scripts execute without preliminary compilation).
- Everyone can use JavaScript without purchasing a license.
- JavaScript is supported by all major browsers, like Netscape and Internet Explorer.

Here's an example:

```
<html>
<body>

<script type="text/javascript">
document.write("Hello World!")
</script>

</body>
</html>
```

It looks like this in the web browser:

Hello World!

VBScript

- VBScript is a scripting language.
- A scripting language is a lightweight programming language.

- VBScript is a light version of Microsoft's programming language Visual Basic.

When a VBScript is inserted into an HTML document, the Internet browser will read the HTML and interpret the VBScript. The VBScript can be executed immediately, or at a later event.

Here's an example:

```
<html>
<body>

<script type="text/vbscript">
document.write("Scripts in the body section are executed when the page is loading")
</script>

</body>
</html>
```

It looks like this in the web browser:

Scripts in the body section are executed when the page is loading

Java Applets

A Java applet is a Java program downloaded from a web server into client browsers and runs inside a Java Virtual Machine (JVM) that is integrated into the web browser. Since they are written in Java, applets have all the features of Java such as platform independence, built-in security mechanism, memory management, error handling, etc. In addition, because Java is a full-featured programming language, applets are nearly as powerful as stand-alone desktop applications. Security is built into Java applets to prevent any writing operation on users' disks. If write access is desired, applets can be digitally signed by vendors and authenticated by third-party authorities to ensure security.

ActiveX Controls

ActiveX technology is an alternative to Java technology for complex web-based UI development. The term ActiveX generally refers to

```
<HTML>
<HEAD>
<TITLE>Applet page : Browser's JVM</TITLE>
</HEAD>
<BODY>
<APPLET CODE=HelloWorld.class ARCHIVE=HelloWorld.jar>
Your browser may not support Java applets
</APPLET>
</BODY>
</HTML>
```

Figure 4(A). Using Browser's JVM

```
<HTML>
<HEAD>
<TITLE>Applet page: Sun's Java Plugin</TITLE>
</HEAD>
<BODY>
<OBJECT classid="clsid:8AD9C840-044E-11D1-
B3E9-00805F499D93"
width=100% height=100
codebase="./j2re-1_3_0_02-win.exe#Version=1,3,0,2">
  <param name="code" value="HelloWorld.class">
  <param name="archive" value="HelloWorld.jar">
  <param name="cache_archive" value="HelloWorld.jar">
  <param name="cache_option" value="Plugin">
</OBJECT>
</BODY>
</HTML>
```

Figure 4(B). Using Sun's Java Plugin

ActiveX controls that are software modules based on Microsoft's component object model (COM) architecture. ActiveX enables a program to add functionality by calling ready-made components that blend in and appear as normal parts of the program, such as toolbars, calculators, and even spreadsheets. The power of ActiveX controls in web development is the fact that they can be linked to a web page and downloaded by any ActiveX-compliant web browser. This virtually turns web pages into software pages that can perform operations just like any desktop application. An ActiveX control can be written in a variety of programming languages from Microsoft's suite of Visual tools, including Visual Studio.Net.

ActiveX controls can be quite dangerous and even lethal to client machines because they make extensive use of the windows operating environment. However, if the application is used in a secure environment such as an intranet, ActiveX can be quite safe. Here's an example of how the tabular data control works. Tabular data control is a

Microsoft ActiveX control that comes pre-installed with all versions of IE4+.

The data file mydata.txt looks like this:

```
Utility|Units
~Electric ~|~KWH~
~Water ~|~KGAL~
~GAS ~|~BTUS~
```

The HTML page is called example1.htm and looks like this:

```
<OBJECT ID="data2" CLASSID="CLSID:333C7BC4-460F-11D0-BC04-
0080C7055A83">
    <PARAM NAME="DataURL" VALUE="mydata.txt">
    <PARAM NAME="UseHeader" VALUE="TRUE">
    <PARAM NAME="TextQualifier" VALUE="~">
    <PARAM NAME="FieldDelim" VALUE="|">
</OBJECT>

<TABLE DATASRC="#data2" BORDER="2">
<THEAD>
    <TH>Utility</TH>
    <TH>Units</TH>
</THEAD>
<TR>
    <TD><SPAN DATAFLD="Utility"></SPAN></TD>
    <TD><SPAN DATAFLD="Units"></SPAN></TD>
</TR>
</TABLE>
```

The following page will display the results as shown below:

[http://utilityreporting.com/activex/example1.htm]

You can read more about the tabular data control at:

[http://wsabstract.com/javatutors/tdc.shtml]

Utility	Units
Electric	kWh
Water	kgal
Gas	Btus

In summary, different client-side technologies have different features and fit different tasks. Client-side scripts may serve as glue to connect HTML, applets, ActiveX controls, and other server-side technologies. While client-side scripts may be the simplest in functionality and easiest to program, applets and ActiveX controls are more complicated and more capable. (16)

CONCLUSION

Not all of the technologies discussed in this paper are required to build effective utility data web pages. There are certainly other technologies not discussed here that can be helpful. Pure HTML is the most reliable method of designing web pages. XHTML is the latest version of HTML, providing the syntax necessary to incorporate XML data into web pages. XML provides portability of data between database formats. CSS makes web pages less complex, more standardized, and produces smaller, faster loading pages. DHTML adds some features that make web pages more intuitive, but add some risk that different web browsers may interpret the page differently. Scripts like JavaScript, VBScript, PHP and ASP add the functionality of a subset of higher level programming languages into web pages. Server-side scripts are more secure but place the processing burden on the server. Therefore, response times may be slow. Client-side scripts are less secure and limited in functionality, but distribute the processing burden across the client machines, leading to faster response times. Java applets are most useful for graphing trends in data. The next article will explain how to use many of these technologies to present utility data in web pages so that the web application is reliable, robust, informative and intuitive.

Acknowledgments:

The authors are grateful for assistance from Paul Allen, chief energy management engineer at Walt Disney World. Paul is responsible for the development and implementation of energy conservation projects throughout the Walt Disney World Resort. Thanks also to Dr. Barney Capehart, Professor Emeritus at the University of Florida College of Engineering and member of the AEE Hall of Fame.

References:

(1) Fangxing Li, Lavelle A.A Freeman and Richard E. Brown, "Web-enable Applications for Outsourced Computing," *IEEE Power and Energy Magazine*, vol. 1, no. 1, pp 53-57, January-February 2003.

(2) "XHTML.ORG" [*article on-line*] (August 4, 2000, accessed 6 May 2003); available from http://www.xhtml.org/; Internet.

(3) Infinite Software Solutions, Inc.,"DevGuru XHTML Introduction" [*article on-line*] (accessed 6 May 2003); available from http://www.devguru.com/Technologies/xhtml/quickref/xhtml_intro.html; Internet.

(4) Richmond, Alan,"DevGuru XHTML Introduction" [*article on-line*] (February 2, 2000, accessed 6 May 2003); available from http://www.wdvl.com/Authoring/Languages/XML/XHTML/; Internet.

(5) Refsnes Data,"XHTML Tutorial" [*article on-line*] (accessed 6 May 2003); available from http://www.w3schools.com/xhtml/; Internet.

(6) W3C,"W3C Markup Validation Service" [*article on-line*] (December 5, 2002, accessed 6 May 2003); available from http://validator.w3.org/; Internet.

(7) Western Civilisation Pty. Ltd.,"The Complete CSS Guide" [*article on-line*] (accessed 6 May 2003); available from http://www.westciv.com/style_master/academy/css_tutorial/toc.html; Internet.

(8) Traversa, Eddie,"Intro to DHTML" [*article on-line*] (September 13, 2000, accessed 6 May 2003); available from http://www.dhtmlshock.com/articles.asp?ArticleID=1; Internet.

(9) " Rough Guide To The DOM" [*article on-line*] (accessed 14 May 2003); available from http://www.devshed.com/Client_Side/DHTML/DOM/DOMpart1/page1.html; Internet.

(10) W3C,"XML-Data" [*article on-line*] (January 5, 1998, accessed 6 May 2003); available from http://www.w3.org/TR/1998/NOTE-XML-data-0105/; Internet.

(11) NCSA,"Common Gateway Interface" [*article on-line*] (accessed 6 May 2003); available from http://hoohoo.ncsa.uiuc.edu/cgi/intro.html; Internet.

(12) Kadow, Kevin,"Common Gateway Interface, Quick Reference Guide" [*article on-line*] (accessed 6 May 2003); available from http://www.msg.net/tutorial/cgi/; Internet.
(13) Sun Microsystems, Inc.,"Java Servlet Technology, The Power Behind the Server" [*article on-line*] (accessed 6 May 2003); available from http://java.sun.com/products/servlet/; Internet.
(14) Sun Microsystems, Inc.,"JavaServer Pages, Dynamically Generated Web Content" [*article on-line*] (accessed 6 May 2003); available from http://java.sun.com/products/jsp/; Internet.
(15) Knoblock, Steve,"Getting Started With PHP" [*article on-line*] (accessed 6 May 2003); available from http://www.phphelp.com/article/1p1.php; Internet.
(16) Fanqxing Li, Lavelle A. A. Freeman, "Using Client-side Technologies to Develop Web-based Applications for Power Distribution Analysis," *Distributech 2003 Conference*, February 4-6, 2003, Las Vegas, Nevada.

ABOUT THE AUTHORS

David C. Green has combined experience in intranet/internet technology and database queries and has developed programming for energy information systems. David has been the president of his own consulting company, Green Management Services, Inc., since 1994. He has a Bachelor of Science degree in chemistry and a Master of Arts degree in computer science. David is also a lieutenant colonel in the Illinois Army National Guard and has 18 years of military service. David has successfully completed major projects for The ABB Group, Cummins Engine Company, ECI Telematics, The M.A.R.C of the Professionals and The Illinois Army National Guard. (dcgreen@dcgreen.com)

Fangxing Li is presently a senior R&D engineering at ABB Inc. He received his B.S. and M.S. degrees both in electric power engineering from Southeastern University, China, in 1994 and 1997 respectively. He received his Ph.D. degree in computer engineering from Virginia Tech in 2001. His main experience includes development of several industry-leading applications for electric power engineering, such as EPRI Distribution Engineering Workstation (DEW) for power distribution planning and analysis, ABB's Power Distribution Optimizer (PDO) for web-based distribution reliability planning, and ABB's GridView for energy market simulation. Dr. Li is a member of IEEE and Sigma Xi. He can be reached at fangxing.li@us.abb.com or fangxing.li@ieee.org.