



CORPORATE IVR

Quickly create applications with ASR, TTS, SQL/ODBC, CTI.

BILLING FOR YOUR IVR

Add real-time call rating and billing capabilities to any IVR.

FROM E-MAIL TO U. MESSAGING

Turn your e-mail servers into a unified messaging platform.

NETWORK IVR & VXML

Multi-services: speech, IP, SS7, conferencing, FAX, VXML, GUI.

www.developer.com/net/asp/article.php/619261

[Back to Article](#)

Generating Bar Graphs with Active Server Pages

By [Don Franke](#)
January 6, 2000

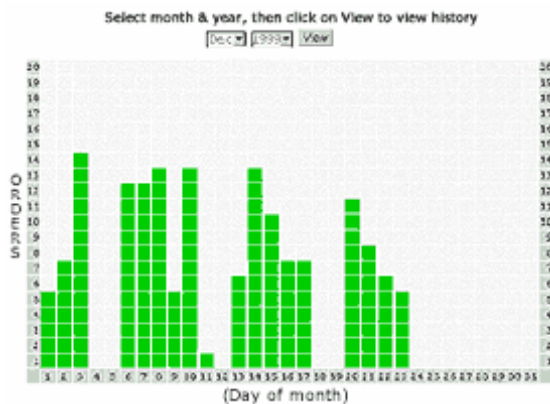


Web pages are visual experiences. When you are presenting numeric data, using a graph is always more visually grabbing, not to

mention quicker for the user to interpret. But using dynamically-generated graphics usually means processing overhead. So you have to decide: should I go graphical and eat the extra processing and download time, or should I keep it fast, text-only, and boring. This article hopefully proposes a happy medium: a dynamic bar graph with only a modest processing delay. How do we accomplish this? After investigating several approaches, I settled on a graph using an HTML table with different-colored cells. The script that makes it all happen is described below.

A Little History

One of the intranet sites I maintain is an ordering site, which provides order history information. The history page must show the total number of orders per day over the course of a month. Previously this data was represented by a two-column table (day number on the left, total orders on the right.) This was pretty dry to look at, and it often took users a moment of staring at the table before they could start to see patterns (i.e. more orders take place later in the week, etc.) So I decided I wanted to cross the textual/graphical divide and see what graphing options were available.



I evaluated several graphing tools. I tested two server-side DLL's that generated GIF graphs. These worked great, but involved registering a DLL, and cost some processing time while the DLL generated the graph. And they cost money. Next I moved to the client side and evaluated a Java

and an ActiveX solution. One worked great out of the gate; the other gave me a little browser compatibility grief I had to iron out. Both applets also took a few seconds to download and generate the graph. Once the graph was displayed, however, it looked great and offered many bells and whistles, like letting the user zoom in, change some display parameters, etc. These applets were also not free, however.

But all I wanted was a bar graph, something a little warmer than a simple text table. I didn't need to have the latest and greatest graphing innovations-they would just be wasted. Nor did I want to slow down the page delivery.

The following is a breakdown of the resulting ASP script I came up with, top to bottom.

The Date

This graph shows the number of orders per day, spread over a specified month. The array `aryMonth` gives an English label to the month number (i.e. 12 is "Dec".) The number of days in the month is retrieved from the array `aryDay`. Both arrays are indexed using the month number. Notice the first element (element 0) of both arrays is empty, since VBScript array indices always begin with zero and there is no month zero.

Next, the data array (`aryData`) is dimensioned to the size of the number of days of the specified month (`aryDay(iMonth)`.) This array will hold the total number of orders for each day.

The Database

This script is used with a database. The array `aryData` is populated by going through each day of the specified month and issuing a database SELECT request for the relevant count. So for December there are 31 database fetches. This is where the most server-side overhead lies, but we do this all at once and put the values into an array. When drawing the graph then we just loop through `aryData`. The data can be stored in a database, a text file, or any other source. Data is data; its source doesn't matter as long as it gets put into `aryData`.

The Graph

Since HTML tables are drawn top-down, and the bar graph is logically drawn top-up, we have to inverse the drawing of the table rows, going from 20 to 1 step -1. The height size of this graph is identified by the variable `iYSize`, and is set to 20 for this example. As a result row 20 is drawn first, and row 1 is last. Since columns are drawn left-right, the column numbers go from the first day to the last day of the month. As a result there are two loops: the row loop and the day loop. Since the table is drawn down, then across, the day of the month loop is nested within the row loop.

```
Loop through rows
  Loop through days
```

The index for the day loop is `y`. In this loop a compare is done to see if the row number (`x`) is greater than the total number of orders for that day (`aryData(y)`.) If the row number is greater than the total count for that day, the cell is light gray; if `x` is equal to or less `aryData(y)`, the cell is green. Example: For row 13, day 7, the value for `aryData(7)` is 12, so the cell is gray. In the next pass of the row loop, though, when the row number is now 12, the cell is green since the value for that day is equal to the row number.

Click here for the [code](#).

Conclusion

The nested loops section is the heart of the code. In this example you will notice I have dressed up the graph with row numbers on either side, and a day number listing along the bottom, as well as the label "Orders" on the side and "Days" along the bottom. There is also a form at the top of the page where the user selects the month and year and clicks "View", which submits these values to itself and refreshes with the corresponding graph.

There is a lot of customization that can be done, like changing the color scheme, using an image instead of cell color to represent a value, etc. You can also rotate the graph 90 degrees by reversing the loops. The bottom line is that any web browser that supports tables supports this graph; no special server-side graphic generators and no client-side applets required. If a basic, tried and true

bar graph is all you need, you might want to consider this solution.

About the author:

Don Franke is a senior developer at Motorola in Schaumburg, Illinois. He has been published and featured in several industry publications, and is currently working on a Masters of Information Systems degree.



JupiterWeb networks:



Search JupiterWeb:

Jupitermedia Corporation has four divisions:
[JupiterWeb](#), [JupiterResearch](#), [JupiterEvents](#) and [JupiterImages](#)

Copyright 2005 Jupitermedia Corporation All Rights Reserved.
[Legal Notices](#), [Licensing](#), [Reprints](#), & [Permissions](#), [Privacy Policy](#).

[Jupitermedia Corporate Info](#) | [Newsletters](#) | [Tech Jobs](#) | [E-mail Offers](#)