Paper 075-2008

Talkin' Bout Code Generation: Getting Animated With SAS® and Excel

Ted Conway, Chicago, IL

ABSTRACT

You've probably seen how SAS® can be used to write SAS code. So why not Excel code? Using SAS to generate Excel VBA code can make lighter work of your "Office" chores. It can also help you bridge those brief gaps between compatible releases of SAS, Excel and Windows, allowing you to exploit new MS-Office features while the folks in Cary address what the folks in Redmond have wrought. In this paper, SAS is used to generate the VBA code behind an animated Excel Chart that presents SAS data using some of the cool tricks available with the copy of Excel 2007 that's sitting on your new Vista Home Basic Edition PC.

INTRODUCTION

Still waiting for The Boss to pop for that new release of JMP you've had your eye on?

Are your pleas for the SAS Microsoft Office Add-In still falling on deaf ears?

Afraid that this means you're doomed to another year of boring pie charts?

Well, think again, Bunky!

Grab a cup of coffee and give us just a few minutes – you'll see how SAS can be used to help generate enough VBA code to produce Excel charts that'll make the folks at USAToday.com green with envy!

PLAY BALL!

To demonstrate this technique, let's turn to the Great American Pastime for inspiration – in a nutshell, our goal is to:

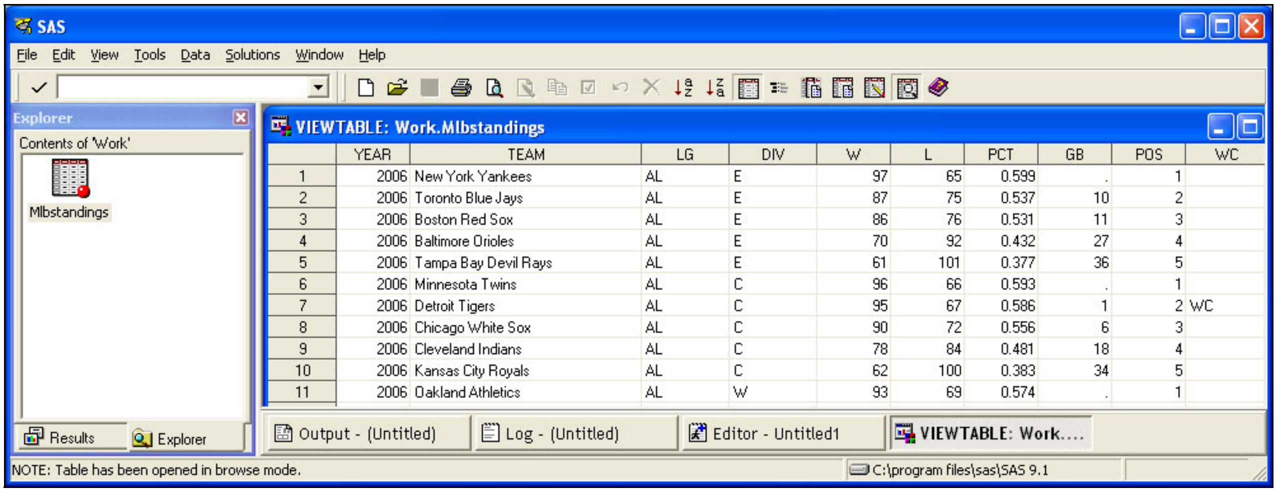
- Present final Major League Baseball standings
- In an easy-to-digest single Excel chart format
- For two Leagues and six Divisions
- With more than thirty+ teams
- Over a ten-year period

Now let's make like Nike and just do it!

DATA, DATA EVERYWHERE

When Tim Berners-Lee invented the World Wide Web, he must've had baseball stats in mind – how else do you explain the widespread proliferation of baseball data that's available on the Net?

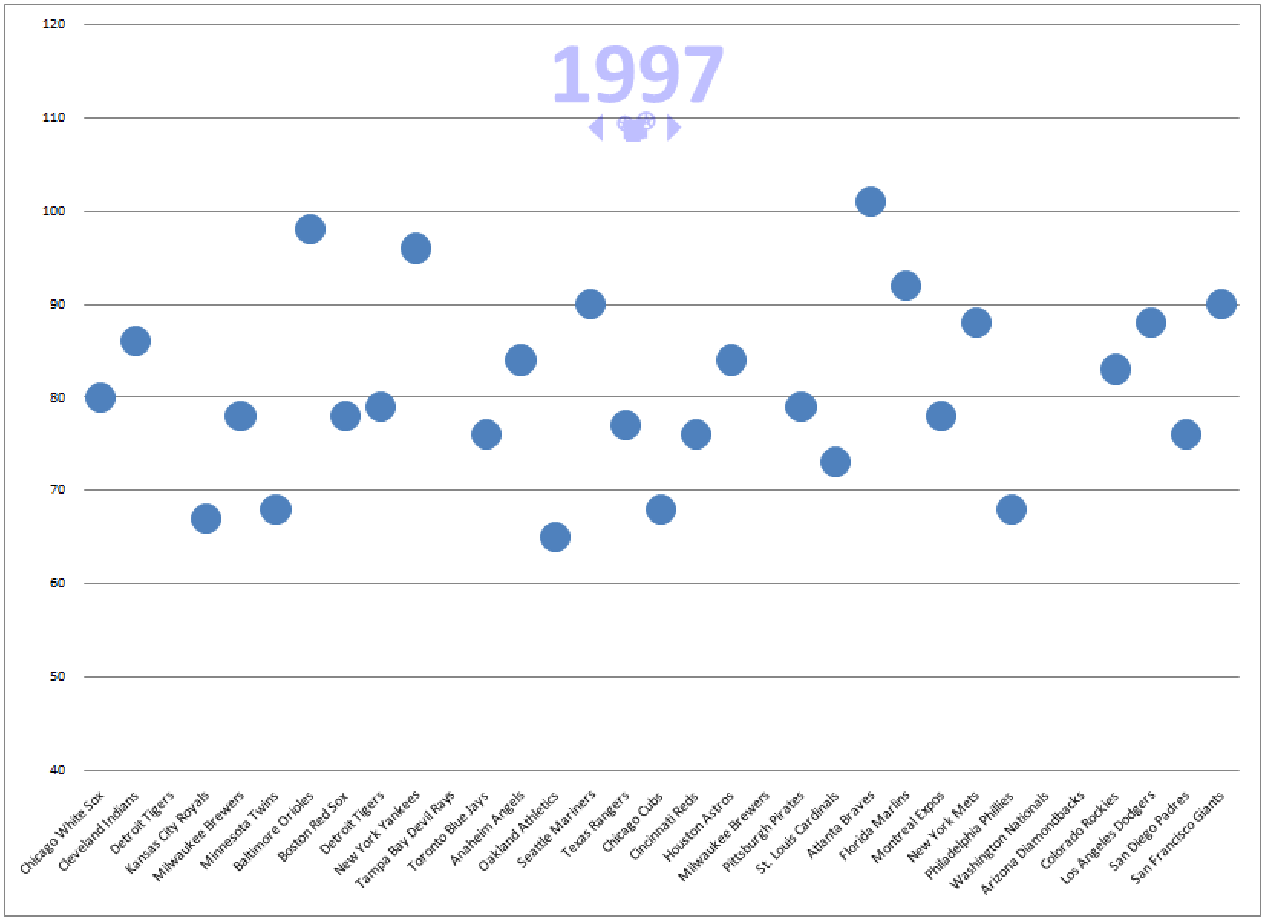
For our purposes, we'll start out by grabbing ten years of final MLB standings from ESPN.com and load them into a SAS dataset – here's a snippet that shows what we've got to work with:



	YEAR	TEAM	LG	DIV	W	L	PCT	GB	POS	WC
1	2006	New York Yankees	AL	E	97	65	0.599	.	1	
2	2006	Toronto Blue Jays	AL	E	87	75	0.537	10	2	
3	2006	Boston Red Sox	AL	E	86	76	0.531	11	3	
4	2006	Baltimore Orioles	AL	E	70	92	0.432	27	4	
5	2006	Tampa Bay Devil Rays	AL	E	61	101	0.377	36	5	
6	2006	Minnesota Twins	AL	C	96	66	0.593	.	1	
7	2006	Detroit Tigers	AL	C	95	67	0.586	1	2	WC
8	2006	Chicago White Sox	AL	C	90	72	0.556	6	3	
9	2006	Cleveland Indians	AL	C	78	84	0.481	18	4	
10	2006	Kansas City Royals	AL	C	62	100	0.383	34	5	
11	2006	Oakland Athletics	AL	W	93	69	0.574	.	1	

NOTE: Table has been opened in browse mode.

ANIMATED CHART – VERSION 1.0 – START SCREEN



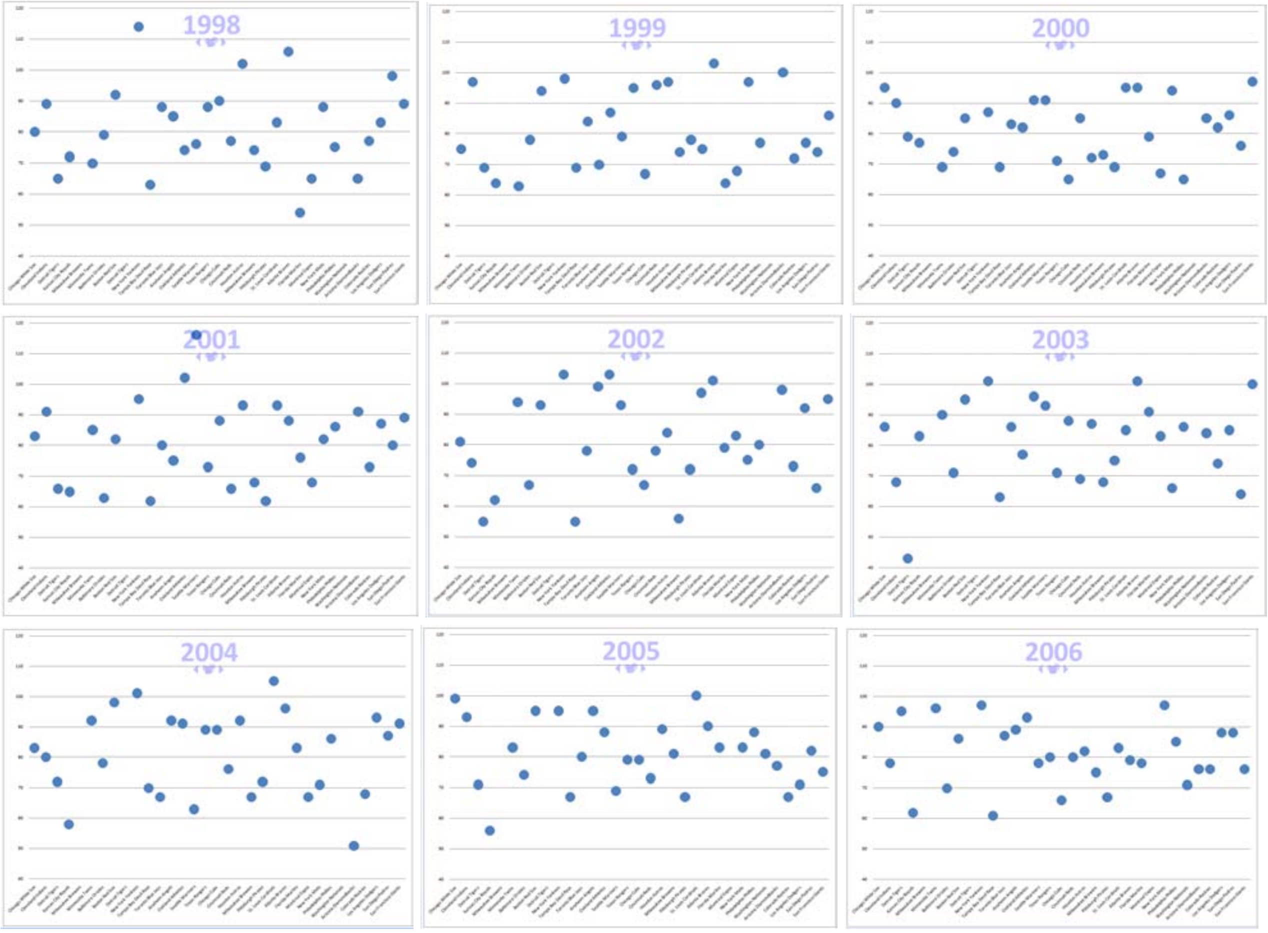
Before delving into the details of the SAS and Excel VBA code, let's take a look at the above chart, which is a first take at where we're going:

- Initially, one is shown the baseball standings for the first year that data is available
- Team names are displayed along the x-axis in the same order as they appear in the data – i.e., by team within league (American or National) and division (East-Central-West)
- The y-axis denotes the number of wins – markers are used to denote the number of wins each team finished the season with
- Controls are provided below the year to allow one to advance forwards or backwards through the standings

1. Clicking on the ***left arrow*** causes data for the previous year to be displayed
2. Clicking on the ***projector*** plays an animated "slideshow" of the standings for all of the years
3. Clicking on the **right arrow** causes data for the previous year to be displayed

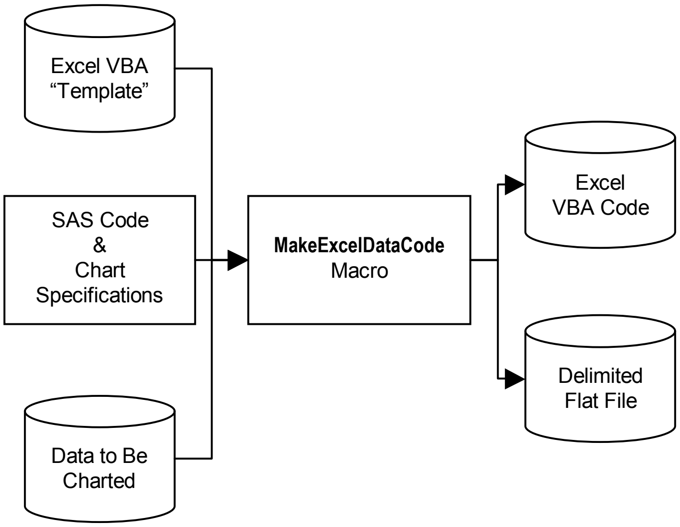
Screenshots of charts for the subsequent years are shown on the following page.

SUBSEQUENT ANIMATED CHARTS



As one uses the controls at the top of the chart to advance through the years, the chart is updated accordingly to reflect each year's final standings, allowing you to see how your favorite team fared against its rivals over the years.

MAKEEXCELDATACODE MACRO – OVERVIEW



1. Create a file with the SAS code you need to ***transform*** the data you wish to chart together with your ***charting specifications***, e.g., data set name, chart variables, scaling.
2. The ***MakeExcelDataCode*** macro uses the parameters you specified together with additional info it gleans from the charting data you provided – e.g., number of rows, year ranges – to properly set macro variables in the ***Excel VBA "Template"*** code. The macro also generates a ***flat file*** of the ***category*** (x-axis), ***value*** (y-axis), and ***time*** (year) variables you identified.
3. From Excel, you can ***load*** (e.g., ALT-F11, IMPORT) and ***run*** the generated VBA code to create your animated chart using the data from the delimited flat fie.

MAKEEXCELDATACODE SAS MACRO

```
%macro MakeExcelDataCode;

*==> Use SAS Data Set to Generate Delimited File for Excel;

options missing=' ';
data _null_;
retain ChartRows ChartMinYr ChartMaxYr;
set &ChartData end=eof;
file "&ChartDir&ChartData..txt";
if _n_=1 then put "&ChartYear|&ChartXvar|&ChartYvar";
put &ChartYear (&ChartXvar &ChartYvar)(+(-1) '|');
chartrows+1;
chartminyr=min(chartminyr,&ChartYear);
chartmaxyr=max(chartmaxyr,&ChartYear);
if eof;
call symput("ChartRows",compress(put(chartrows+1,7.)));
call symput("ChartMinYr",compress(put(chartminyr,7.)));
call symput("ChartMaxYr",compress(put(chartmaxyr,7.)));

*==> Use Template File + Substitution to Generate Excel VBA Code;

data _null_;
infile "&ChartDir.ExcelVBAtemplate.txt" lrecl=255;
input;
file "&ChartDir&ChartData.VBACODE.txt" lrecl=255;
_infile_=resolve(_infile_);
put _infile_;
%mend;
```

SAS CODE AND SPECIFICATIONS FOR MLB CHART

```
*==> Summarize Data for Charting;

data mlb;
infile 'c:\users\ted\documents\sgf2008\MLB1997to2006.txt' dlm='|' dsd;
input @;
if _n_>1;
input Year Team : $30. League : $2. Division : $1. Wins;

*==> Create All Possible Combos of Keys and Sort;

proc sql;
create table mlbdummy as
select Team, League, Division, Year, . as Wins from
  (select distinct Team, League, Division from mlb) t1, (select distinct Year from mlb) t2;

create table MLBstandings as
select Team, League, Division, Year, sum(Wins) as Wins from (
  select Team, League, Division, Year, Wins from mlb union all
  select Team, League, Division, Year, Wins from mlbdummy) t1
group by 1, 2, 3, 4 order by 2, 3, 1, 4;

*==> Use Macro to Generate Data (Flat File) and VBA Code For Excel;

%let ChartData=MLBstandings;
%let ChartYear=Year; %let ChartXvar=Team; %let ChartYvar=Wins;
%let ChartMinScale=40; %let ChartMaxScale=120; %let ChartMajorUnit=10;
%let ChartDir=C:\Users\TED\Documents\sgf2008\;

%MakeExcelDataCode;
```

EXCEL VBA CODE TEMPLATE

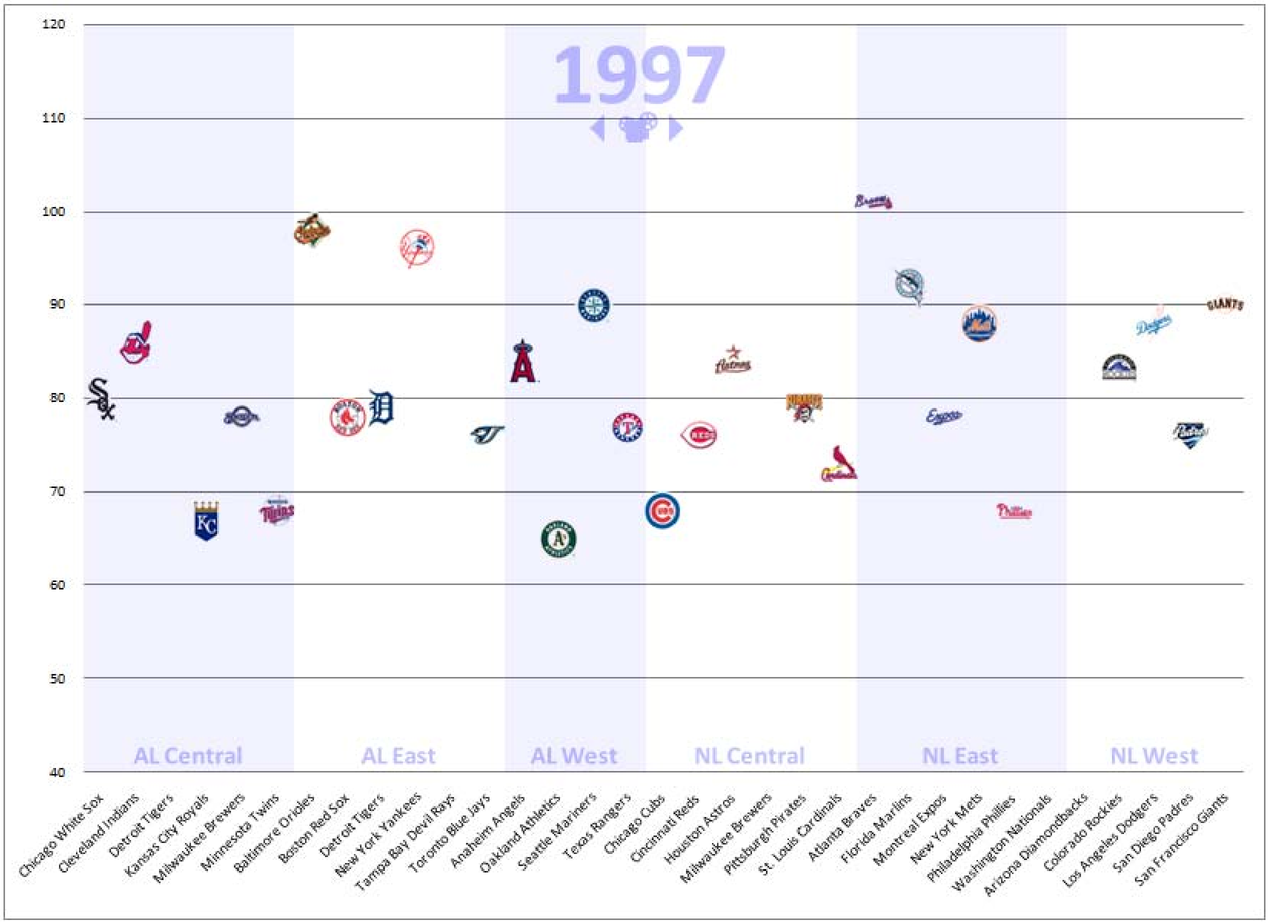
```
Sub MacroGenerateWorkbook()

'==> Import Data, Set Up Year Filter, Do Basic Chart Formatting

Workbooks.OpenText Filename:="&ChartDir&ChartData..txt", _
  DataType:=xlDelimited, Other:=True, OtherChar:="|"
Columns("A:A").Select
Selection.AutoFilter
ActiveSheet.Range("$A$2:$A$&ChartRows").AutoFilter Field:=1, Criteria1:="&ChartMinYr"
ActiveSheet.Shapes.AddChart.Select
ActiveChart.ChartType = xlLineMarkers
ActiveChart.Location Where:=xlLocationAsNewSheet
ActiveChart.SetSourceData Source:=Sheets("&ChartData").Range("A1:C&ChartRows")
ActiveChart.SeriesCollection(1).XValues = "=&ChartData!$B$2:$B$&ChartRows"
ActiveChart.SeriesCollection(1).Values = "=&ChartData!$C$2:$C$&ChartRows"
ActiveChart.SeriesCollection(1).Name = "&ChartYvar"
ActiveChart.HasTitle = False
ActiveChart.Legend.Delete
ActiveChart.Axes(xlValue).Border.ColorIndex = xlNone
ActiveChart.Axes(xlValue).MinimumScale = &ChartMinScale
ActiveChart.Axes(xlValue).MaximumScale = &ChartMaxScale
ActiveChart.Axes(xlValue).MajorUnit = &ChartMajorUnit
ActiveChart.Axes(xlValue).TickLabels.Font.Size = 8
ActiveChart.Axes(xlCategory).TickLabels.Font.Size = 8
ActiveChart.Axes(xlCategory).MajorTickMark = xlNone
ActiveChart.SeriesCollection(1).MarkerStyle = xlMarkerStyleCircle
ActiveChart.SeriesCollection(1).MarkerSize = 16
ActiveChart.SeriesCollection(1).Border.ColorIndex = xlNone

'==> Define Text Box to Hold Year of Data Being Displayed

ActiveChart.Shapes.AddTextbox msoTextOrientationHorizontal, 100, 0, 250, 48
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.Text = "&ChartMinYr"
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.FontSize = 48
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.FontBold = msoTrue
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextFrame.Characters.Font.Color = RGB(0, 0, 255) ' (Blue)
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextFrame2.TextRange.Font.Fill.Transparency = 0.75
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.Alignment = msoTextEffectAlignmentCentered
ActiveChart.Shapes(ActiveChart.Shapes.Count).Left = ActiveChart.ChartArea.Width / 2 - _
  ActiveChart.Shapes(ActiveChart.Shapes.Count).Width / 2
ActiveChart.Shapes(ActiveChart.Shapes.Count).Name = "YR"

'==> Define Controls to Advance Year Values - Previous (Left Arrow), All (Projector), Next(Right Arrow)

ActiveChart.Shapes.AddTextbox msoTextOrientationHorizontal, Sheets("Chart1").Shapes("YR").Left, _
  Sheets("Chart1").Shapes("YR").Top + Sheets("Chart1").Shapes("YR").Height, Sheets("Chart1").Shapes("YR").Width, 24
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.Text = Chr$(184) ' (Projector)
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.FontName = "Webdings"
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.FontSize = 24
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.FontBold = msoTrue
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextFrame.Characters.Font.Color = RGB(0, 0, 255) ' (Blue)
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextFrame2.TextRange.Font.Fill.Transparency = 0.75
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.Alignment = msoTextEffectAlignmentCentered
ActiveChart.Shapes(ActiveChart.Shapes.Count).OnAction = "MacroAll"
ActiveChart.Shapes.AddTextbox msoTextOrientationHorizontal, Sheets("Chart1").Shapes("YR").Left - 20, _
  Sheets("Chart1").Shapes("YR").Top + Sheets("Chart1").Shapes("YR").Height, Sheets("Chart1").Shapes("YR").Width, 24
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.Text = Chr$(51) ' (Left Arrow)
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.FontName = "Webdings"
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.FontSize = 24
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.FontBold = msoTrue
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextFrame.Characters.Font.Color = RGB(0, 0, 255) ' (Blue)
```

EXCEL VBA CODE TEMPLATE (CONTINUED)

```
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextFrame2.TextRange.Font.Fill.Transparency = 0.75
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.Alignment = msoTextEffectAlignmentCentered
ActiveChart.Shapes(ActiveChart.Shapes.Count).OnAction = "MacroPrev"
ActiveChart.Shapes.AddTextbox msoTextOrientationHorizontal, Sheets("Chart1").Shapes("YR").Left + 20, _
  Sheets("Chart1").Shapes("YR").Top + Sheets("Chart1").Shapes("YR").Height, Sheets("Chart1").Shapes("YR").Width, 24
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.Text = Chr$(52) ' (Right Arrow)
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.FontName = "Webdings"
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.FontSize = 24
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.FontBold = msoTrue
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextFrame.Characters.Font.Color = RGB(0, 0, 255) ' (Blue)
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextFrame2.TextRange.Font.Fill.Transparency = 0.75
ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.Alignment = msoTextEffectAlignmentCentered
ActiveChart.Shapes(ActiveChart.Shapes.Count).OnAction = "MacroNext"

'==> Copy Routines to Advance Years

Workbooks("&ChartData..txt").VBProject.VBComponents.Import "&ChartDir&ChartData.VBACODE.txt"

ActiveChart.ChartArea.Select
ActiveWindow.Zoom = 100
Application.DisplayFullScreen = True
End Sub

'==> Display Data for All Years

Sub MacroAll()
For y = &ChartMinYr To &ChartMaxYr
  Sheets("&ChartData").Range("$A$2:$A$&ChartRows").AutoFilter Field:=1, Criteria1:=Format(y, "####")
  Sheets("Chart1").Shapes("YR").TextEffect.Text = Format(y, "####")
  Stoptime = Timer + 0.5   ' (Delay)
  Do While Timer < Stoptime
    DoEvents    ' (Yield to other processes)
  Loop
Next
End Sub

'==> Display Data for Next Year

Sub MacroNext()
y = Val(Sheets("Chart1").Shapes("YR").TextEffect.Text) + 1
If y > &ChartMaxYr Then y = &ChartMinYr
Sheets("&ChartData").Range("$A$2:$A$&ChartRows").AutoFilter Field:=1, Criteria1:=Format(y, "####")
Sheets("Chart1").Shapes("YR").TextEffect.Text = Format(y, "####")
End Sub

'==> Display Data for Previous Year

Sub MacroPrev()
y = Val(Sheets("Chart1").Shapes("YR").TextEffect.Text) - 1
If y < &ChartMinYr Then y = &ChartMaxYr
Sheets("&ChartData").Range("$A$2:$A$&ChartRows").AutoFilter Field:=1, Criteria1:=Format(y, "####")
Sheets("Chart1").Shapes("YR").TextEffect.Text = Format(y, "####")
End Sub
```

Note: SAS macro variables will be resolved by the ***MAKEEXCELDATACODE*** macro.

ANIMATED CHART – VERSION 2.0 – START SCREEN



The approach described in the previous pages is attractive in that it's generic – it will work fairly well with any data that falls into the macro's simple three variable-constraint (i.e., X, Y and YEAR), not just the MLB data.

Like all software, however, there are always some tradeoffs between a generic and semi-custom solution.

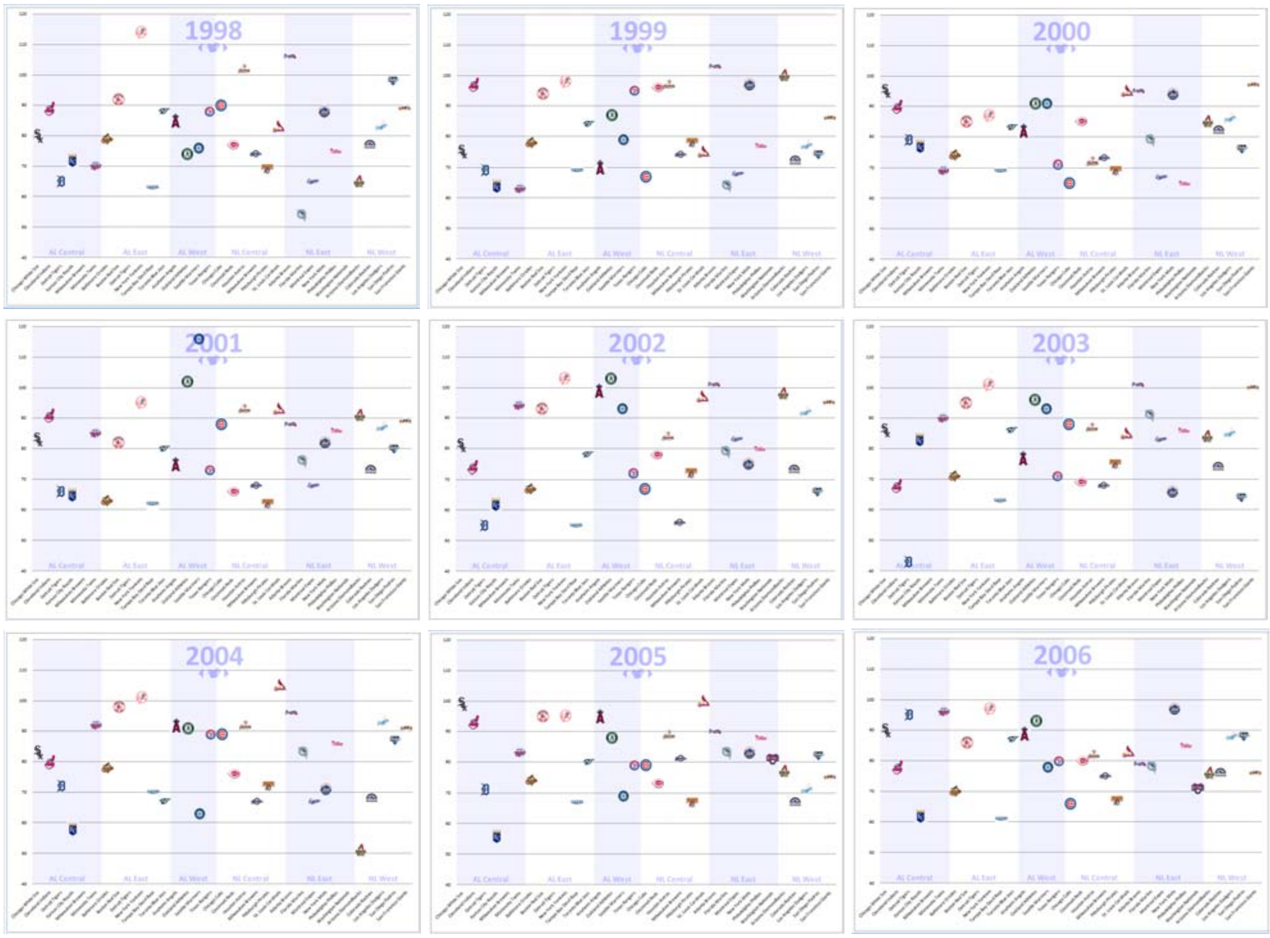
And in this case, the revised chart shown above illustrates how a little custom-VBA tweaking can help us out immensely in a couple of problem areas with the original chart:

1. **Markers** - While the teams are nicely labeled at the bottom of the screen, it can be difficult to make a connection between the team labels at the bottom of the screen and the generic markers above. To address this, the revised chart replaces the generic markers with ***markers corresponding to the teams' MLB logos***. Major League Baseball makes artwork freely available at ***http://pressbox.mlb.com/pressbox/logos/index.jsp***. The images can be simply cut-and-pasted from the ***mlb.com*** web pages into an Excel worksheet, after which they can be programmatically manipulated to create the custom MLB logo markers.

2. **Categories** -While the teams are grouped together by their league and division (some teams appear twice due to changes over the years), it's also not easy to determine where one division starts and where another begins. Coding was initially done to dynamically change the left-to-right order of teams each season to reflect the order in which they finished in their division, but this approach had to be abandoned due to flickering and the other performance challenges. As an alternative, ***semi-transparent shapes in alternating colors*** were used to highlight the teams making up each division and ***category labels*** were added just above the x-axis to clearly identify each team's leagues/division membership.

The custom VBA code added to effect the changes resulting in the above chart are provided on the following pages together with screenshots of the MLB logo versions of the subsequent years' animated charts.

SUBSEQUENT ANIMATED CHARTS – VERSION 2.0



CUSTOM VBA CODE TO ADD MLB LOGO MARKERS

```
'==> Replace Markers with Easier-to-Identify Team Logos

Workbooks.Open Filename:="C:\Users\TED\Documents\sgf2008\MLBlogos.xlsx"
teams = Array("Baltimore Orioles", "Chicago White Sox", "Anaheim Angels", "Atlanta Braves", "Chicago Cubs", _
          "Arizona Diamondbacks", "Boston Red Sox", "Cleveland Indians", "Oakland Athletics", "Florida Marlins", _
          "Cincinnati Reds", "Colorado Rockies", "New York Yankees", "Detroit Tigers", "Seattle Mariners", _
          "Philadelphia Phillies", "Houston Astros", "Los Angeles Dodgers", "Tampa Bay Devil Rays", _
          "Kansas City Royals", "Texas Rangers", "New York Mets", "Milwaukee Brewers", "San Diego Padres", _
          "Toronto Blue Jays", "Minnesota Twins", "Washington Nationals", "Pittsburgh Pirates", _
          "San Francisco Giants", "St. Louis Cardinals", "Montreal Expos")
Workbooks("MLBstandings.txt").Activate
xteams = ActiveChart.SeriesCollection(1).XValues
For p = 1 To ActiveChart.SeriesCollection(1).Points.Count
For p2 = 0 To UBound(teams)
If xteams(p) = teams(p2) Then
   Workbooks("MLBlogos.xlsx").Sheets("Sheet1").Pictures(p2 + 1).Copy
   ActiveChart.SeriesCollection(1).Points(p).Select
   ActiveChart.SeriesCollection(1).Points(p).Paste
End If
Next
Next
Workbooks("MLBlogos.xlsx").Activate
ActiveWindow.Close
```

Note: To maintain proper z-order, add custom VBA code routines after code that defines a text box to hold the year.

CUSTOM VBA CODE TO HIGHLIGHT LEAGUE/DIVISION MEMBERSHIP

```
'==> Add Shading and Labels to Highlight Leagues and Division Membership

div = Array("AL Central", "AL East", "AL West", "NL Central", "NL East", "NL West")
divCount = Array(6, 6, 4, 6, 6, 5) ' (# of Teams in Each League/Division)
nxtLeft = ActiveChart.Axes(xlCategory).Left
For i = 0 To UBound(div)
 ActiveChart.Shapes.AddTextbox msoTextOrientationHorizontal, nxtLeft, ActiveChart.Axes(xlCategory).Top - 22, _
 ActiveChart.Axes(xlCategory).Width * divCount(i) / 33, 22
 ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.Text = div(i)
 ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.FontSize = 14
 ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.FontBold = msoTrue
 ActiveChart.Shapes(ActiveChart.Shapes.Count).TextFrame.Characters.Font.Color = RGB(0, 0, 255) ' (Blue)
 ActiveChart.Shapes(ActiveChart.Shapes.Count).TextFrame2.TextRange.Font.Fill.Transparency = 0.75
 ActiveChart.Shapes(ActiveChart.Shapes.Count).TextEffect.Alignment = msoTextEffectAlignmentCentered
 ActiveChart.Shapes.AddTextbox msoTextOrientationHorizontal, nxtLeft, ActiveChart.Axes(xlValue).Top, _
 ActiveChart.Axes(xlCategory).Width * divCount(i) / 33, ActiveChart.Axes(xlValue).Height
 If i Mod 2 = 0 Then
  ActiveChart.Shapes(ActiveChart.Shapes.Count).Fill.BackColor.RGB = RGB(0, 0, 255) ' (Blue)
  ActiveChart.Shapes(ActiveChart.Shapes.Count).Fill.Transparency = 0.95
 End If
nxtLeft = nxtLeft + ActiveChart.Shapes(ActiveChart.Shapes.Count).Width
Nextput _infile_;
%mend;
```

CONCLUSION AND LESSONS LEARNED

With a little elbow-grease, SAS can be used to transform data and generate complicated VBA code to make it easier for you to produce visually-appealing Excel 2007 charts with some rudimentary animation, regardless of which platform SAS and your data reside on.

There were also some other important lessons learned from this little project, including:

☺ There's a good reason why people shell out their hard-earned cash to buy software like SAS/JMP – you simply aren't going to be able to build sophisticated bubble chart animations with features like trails, drill-downs, multi-condition filtering, and smooth continuous scrolling of graphics using a do-it-yourself MS-Office 2007 VBA approach!

☺ Never try to do MS-Excel 2007 graphics work on a $399, 512K, one-processor, pre-SP1 Vista Home Basic Edition target machine!

☺ There are also many good reasons why software vendors can't always be ready Day 1 for new Microsoft software releases!

☺ Don't fool yourself into thinking you'll be able to meet a tight paper deadline using MS-Word 2007 if you're not already familiar with the radically different user interface!

REFERENCES AND ACKNOWLEDGEMENTS

Be sure to check out data visualization guru Stephen Few's excellent White Paper on ***Visualizing Change: An Innovation in Time-Series Analysis***, which you'll find at:

http://www.perceptualedge.com/articles/visual_business_intelligence/visualizing_change.pdf

An older-but-still-excellent SUGI 30 paper by Mike Zdeb and Robert Allison, ***Stretching the Bounds of SAS/GRAPH Software***, is also well worth the read – it can be found at:

http://www2.sas.com/proceedings/sugi30/137-30.pdf

CONTACT INFORMATION

Ted Conway resides in Chicago, Illinois. He can be reached at tedconway@aol.com.

TRADEMARKS