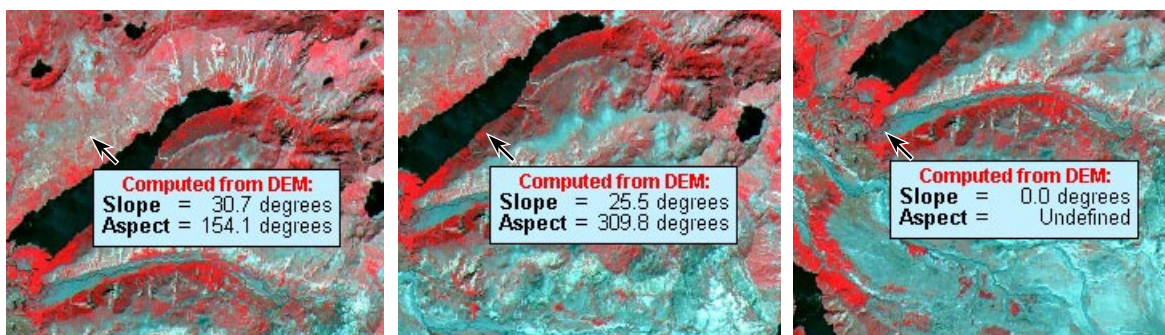


Enhanced DataTips and GraphTips

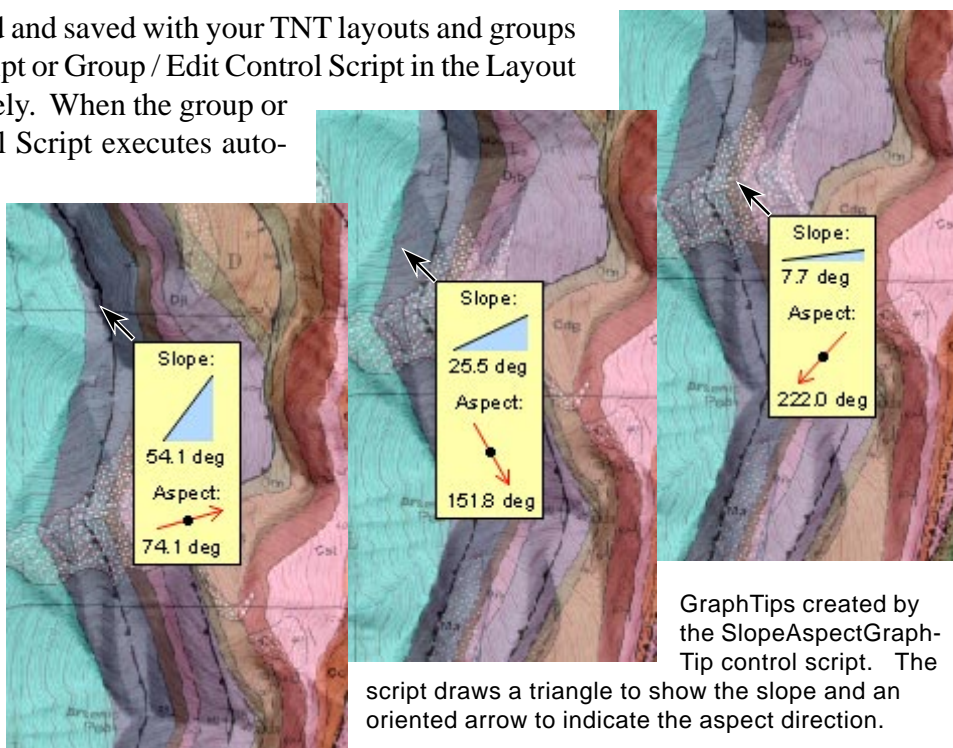
Standard DataTips automatically pop into view to show information (in text form) about the cursor location from the geodata layers in the View window. With a Display Control Script you can create more informative *Enhanced DataTips* and *GraphTips* that pop in with customized or computed content. A Display Control Script can access the cursor position as the DataTip is about to appear, use this position to obtain data from one or more layers in the view (raster cell value, database attribute value for the enclosing polygon, and so on), and present the data or results computed from it. To create an Enhanced DataTip, the script restyles, adds to, and/or replaces the standard DataTip text to provide custom text information. In a GraphTip, the script presents the information primarily in graphical form using class methods to draw graphical elements and text. A Display Control Script can transform the DataTip into a powerful information center that automatically presents a wealth of dynamically-changing data every time the user of your data moves the mouse cursor.

As an example, consider a group or layout that includes an elevation raster. A standard DataTip can report the elevation value for the raster cell under the cursor (along with information from other layers in the view). The sample Enhanced DataTip and GraphTip illustrated here use the elevation raster to compute the local terrain slope and slope direction (aspect) at the cursor position and pop-in the result. The Enhanced DataTip presents the results as labeled and formatted text, whereas the GraphTip combines graphics and text. Excerpts from the Display Control Script that creates the GraphTip are shown on the opposite side of this plate, and both scripts are available for download from www.microimages.com/freestuf/scripts.htm.



Enhanced DataTip created by the SlopeAspect DataTip script. Since aspect is undefined for flat areas, the script does not compute aspect if slope = 0, and instead reports aspect as "Undefined".

Display Control Scripts can be created and saved with your TNT layouts and groups by selecting Layout / Edit Control Script or Group / Edit Control Script in the Layout or Group Controls window, respectively. When the group or layout is opened, the Display Control Script executes automatically to activate the Enhanced DataTip or GraphTip, which then is presented automatically whenever the mouse cursor pauses over the view. And when you include the layout in an atlas, the Enhanced DataTip or GraphTip automatically appears in the TNTatlas window. You can therefore use Display Control Scripts to add custom data exploration capability to the geodata sets you prepare for clients or other end users, capability that is activated automatically and requires no selection or set-up by them.



GraphTips created by the SlopeAspectGraphTip control script. The script draws a triangle to show the slope and an oriented arrow to indicate the aspect direction.

Many sample scripts have been prepared to illustrate how you might use the features of the TNT products' scripting language for scripts and queries. These scripts can be downloaded from www.microimages.com/freestuf/scripts.htm.

Script Excerpt for Slope-Aspect GraphTip

```

proc OnGroupCreateView (
  class GRE_GROUP group
) {
  DEM_layer = group.FirstLayer;
  DispGetRasterFromLayer(DEM, DEM_layer);
  w = ColScale(DEM);
  h = LinScale(DEM);
  nullDEM = NullValue(DEM);
}

func OnViewDataTipShowRequest (
  class GRE_VIEW view,
  class POINT2D point,
  class TOOLTIP datatip
) {
  numeric retval = 1;

  trans = view.GetTransLayerToScreen(DEM_layer, 1);
  ptLayer = trans.ConvertPoint2DFwd(point);

  lin = floor(ptLayer.y);
  col = floor(ptLayer.x);

  if ( DEM[lin,col] <> nullDEM ) {

    dzX = ( DEM[lin, col + 1] - DEM[lin, col - 1] ) / ( 2 * w );
    dzY = ( DEM[lin - 1, col] - DEM[lin + 1, col] ) / ( 2 * h );

    slope = atan( sqrt( sqr(dzX) + sqr(dzY) ) );

    baseline = 40 * cosd(slope);
    triHeight = baseline * tand(slope);
    tribaseY = 17 + round(triHeight);

    aspect$ = "Undefined";
    aspHeight = 5;

    if ( dzY != 0 ) {
      aspect = deg * atan2(-dzX, -dzY);

      if ( aspect < 0 ) then
        aspect += 360;
      else
        aspect = 90;

      aspect$ = sprintf("%.1f deg", aspect);
      aspectDrawAngle = ( aspect - 90 );

      aspHeight = round( abs( 18 * sind(aspectDrawAngle) ) );
      if ( aspHeight < 5 ) then
        aspHeight = 5;

      width = 54;
      center.x = width / 2;
      center.y = tribaseY + 35 + aspHeight;
      height = center.y + aspHeight + 18;

      imagedev.Create(height,width);
      maskdev.Create(height,width);
    }
  }
}

```

```

begin drawing GraphTip elements
gc = imagedev.CreateGC();
gc.SetColorRGB(100,100,67,100);
gc.FillRect(0, 0, width, height);
gc.SetColorName("black");
gc.DrawRect(0, 0, width - 1, height - 1);
gc.DrawTextSetFont("ARIAL.TTF");
gc.DrawTextSetHeightPixels(10);
gc.DrawTextSimple("Slope:", 12, 12);
gc.DrawTextSimple("Aspect:", 10, tribaseY + 30);

x_points[1] = center.x - baseline / 2;
x_points[2] = center.x + baseline / 2;
y_points[1] = tribaseY;
y_points[2] = tribaseY;

if ( slope < 5 ) {
  gc.MoveTo(x_points[1], y_points[1]);
  gc.DrawTo(x_points[2], y_points[2]);
}

else {
  x_points[3] = center.x + baseline / 2;
  y_points[3] = tribaseY - triHeight;

  gc.SetColorRGB(70, 85, 100, 100);
  gc.DrawPolyLine(x_points, y_points, 3);
  gc.FillPolyLine(x_points, y_points, 3);

  gc.MoveTo(x_points[1], y_points[1]);
  gc.SetColorRGB(0, 0, 0, 100);
  gc.DrawTo(x_points[3], y_points[3]);

  gc.DrawTextSimple( sprintf("%.1f deg", slope), 7, tribaseY + 12);

  if ( slope > 0 ) {
    arrowEnd.x = center.x + 18 * cosd(aspectDrawAngle);
    arrowEnd.y = center.y + 18 * sind(aspectDrawAngle);

    gc.MoveTo( center.x - 16 * cosd(aspectDrawAngle), center.y - 16 * sind(aspectDrawAngle) );
    gc.SetColorRGB(100,0,0,100);
    gc.DrawTo(arrowEnd.x, arrowEnd.y );

    gc.DrawArrow(arrowEnd.x, arrowEnd.y, aspectDrawAngle, 6, 30, "Open");

    gc.SetColorRGB(0,0,0,100);
    gc.FillCircle(center.x, center.y, 2);

    gc.DrawTextSimple(aspect$, 5, center.y + aspHeight + 12);

    datatip.SetImageTip(imagedev, maskdev, offset);
  }

  else
    retval = -1;

  return (retval);
}

```