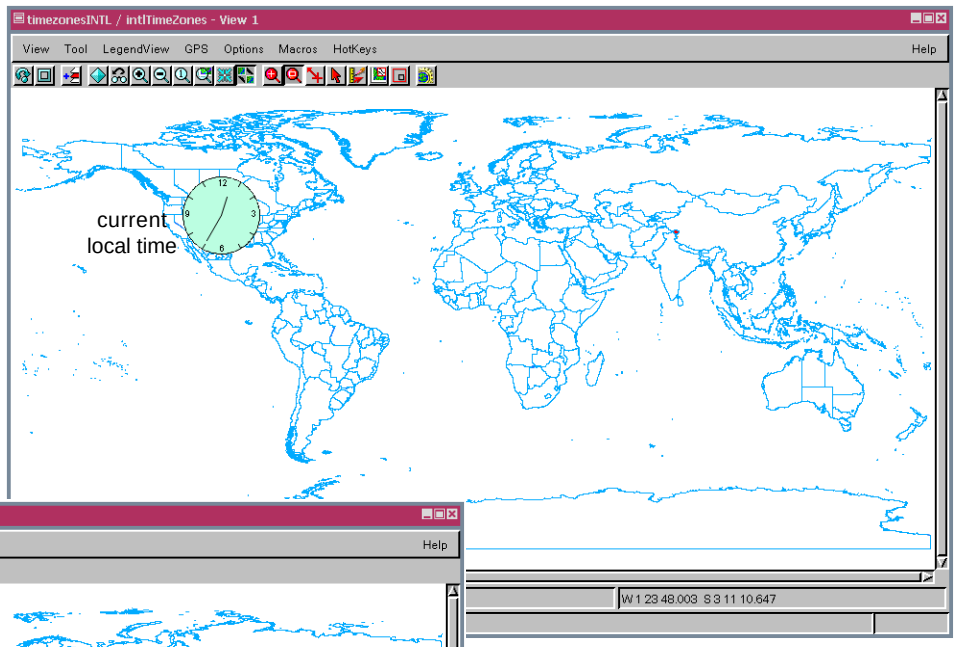
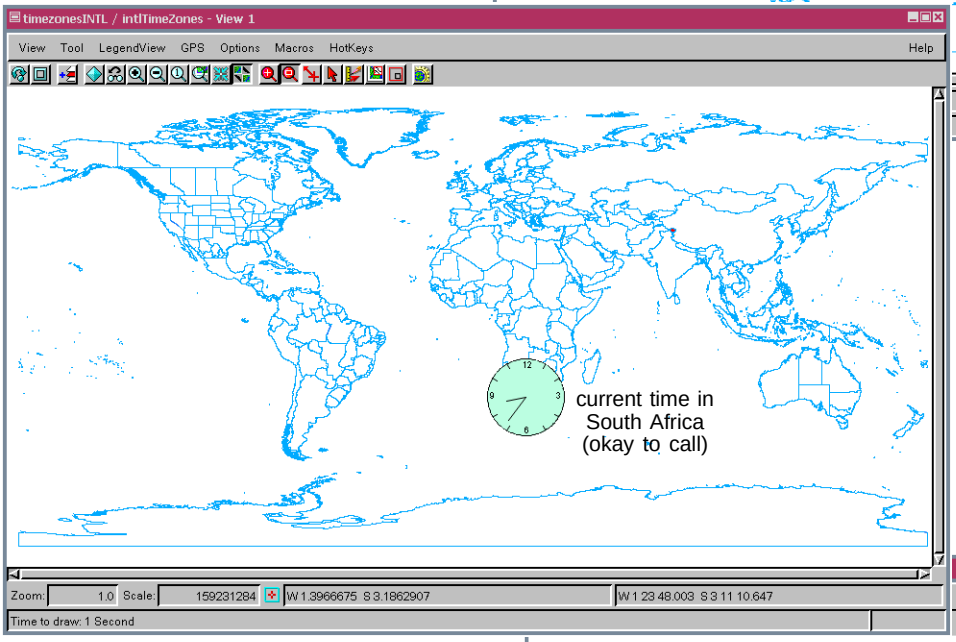
Sample GraphTip Script

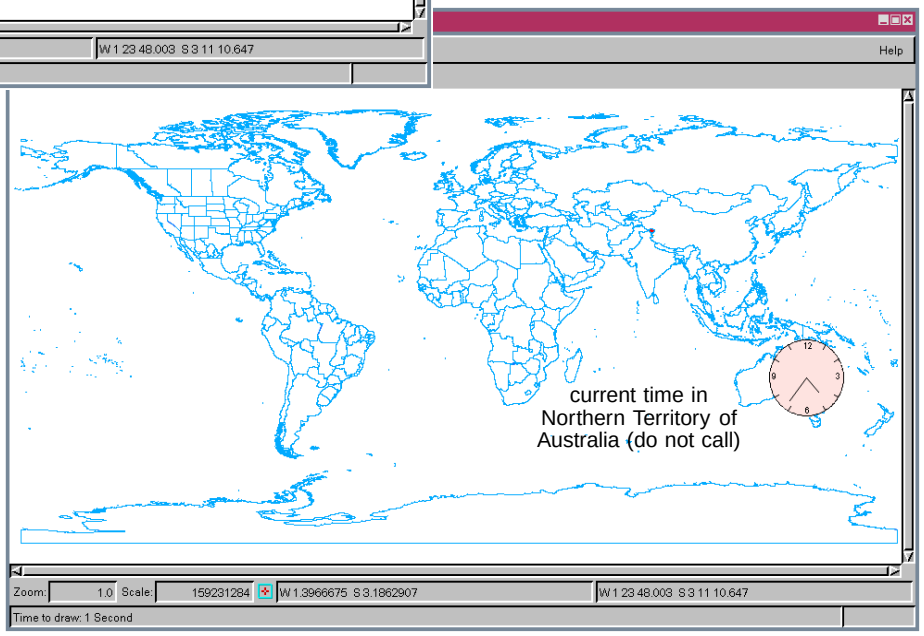
Local Time Zones

GraphTips can pop in dynamically changing information associated with the geographical information in a view. This simple example determines the local time from the host computer and dynamically associates it with the location of the geographic position of the cursor. This GraphTip is easy to test since the local time used in the script is automatically available on any computer platform. Following this example, scripts can be designed that automatically associate the cursor position with sensor networks, local or remote programs, URLs, linked databases, and other external sources.





The sample script shown on the back of this page uses the local time set on your computer, your timezone offset from Greenwich Meridian Time (GMT) obtained from a database field, and the offset from your time zone of the closest polygon to the cursor location over the map being displayed to determine the time shown in the clock graphic. This latter value is obtained from a virtual field that computes the time zone offset between your time zone and any time zone of interest. The time zone map is a hidden layer in the group used for illustration. The clock face could be color coded to reflect whether the time shown is a.m. or p.m. but in the sample reflects acceptable calling times (after 8 a.m. and before 10 p.m.).



These time zone illustrations all use a global map. You can substitute more detailed maps for larger scale displays without altering your time zone map or the control script for a group/layout that contains more detailed, map scale controlled maps.

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 for Time Zone Reporting (timezone.sml)

```
class GRDEVICE_MEM_BINARY maskdev;
class GRDEVICE_MEM_RGB24 imagedev;
class GC gc;
class POINT2D offset;
class GRE_LAYER_VECTOR  timezone_layer;
class VECTOR Vect;
class GRE_GROUP group;
class TRANSPARM trans;
class POINT2D cursorPt;
class GEOREF georef;
class COLOR color;

proc OnInitialize () {
   imagedev.Create(100,100);
   maskdev.Create(100,100);
   gc = maskdev.CreateGC();
   gc.SetColorPixel(1);
   gc.FillCircle(50,50,40);
   offset.x = -50;
   offset.y = -50;
}

proc OnGroupCreateView (class GRE_GROUP group) {
   timezone_layer =
        (class GRE_LAYER_VECTOR)group.FirstLayer;
   DispGetVectorFromLayer (Vect,timezone_layer);
   georef = GetLastUsedGeorefObject(Vect);
}

func OnViewDataTipShowRequest (
        class GRE_VIEW view,
        class POINT2D point,
        class TOOLTIP datatip
        ) {
trans = view.GetTransLayerToScreen(timezone_layer, 1);
cursorPt = trans.ConvertPoint2DFwd (point);

closestPoly = FindClosestPoly(Vect,cursorPt.x, cursorPt.y,
         georef);
local class DATETIME now;
now.SetCurrent();

currentHour = now.GetHour();
currentMin = now.GetMin();

x=Vect.poly[closestPoly].timeznp020[1].Central_OffsetH;
minOffset = (x % 1) * 60;
hourOffset = int(x);
if ((currentMin + minOffset) >= 60)
hourOffset = hourOffset + 1;
modifiedMin = (currentMin + minOffset) % 60;
modifiedHour = (hourOffset + currentHour) ;

gc = imagedev.CreateGC();

if (((modifiedHour%24) >=22) or ((modifiedHour%24)
         < 8))
gc.SetColorName("misty rose");
else gc.SetColorName("sea foam");
gc.FillCircle(50,50, 40);
gc.SetColorName("black");
gc.DrawCircle(50,50,40);
gc.MoveTo(50,50);

local numeric h = ((modifiedHour% 12) + modifiedMin/
        60) * 30 - 90;
local numeric m = modifiedMin * 6 - 90;
gc.DrawTo(20*cosd(h) + 50 , 20*sind(h) + 50);
gc.MoveTo(50,50);
gc.DrawTo(30*cosd(m) + 50, 30*sind(m) + 50);

gc.DrawTextSetFont("ARIAL.TTF");
gc.DrawTextSetHeightPixels(9);
color.Name="black";
gc.SetColor(color);
gc.DrawTextSetColors(color);
gc.DrawTextSimple("12",47,20);
gc.DrawTextSimple("6",49,88);
gc.DrawTextSimple("3",82,52);
gc.DrawTextSimple("9",13,52);

numeric tick1, tick2, tick4, tick5, tick7, tick8, tick10,
tick11;
tick1=1 * 30 - 90;

gc.MoveTo(50,50);
gc.MoveTo(35 *cosd(tick1) + 50, 35*sind(tick1) + 50);
gc.DrawTo(50 *cosd(tick1) + 50, 50*sind(tick1) + 50);

datatip.SetImageTip(imagedev, maskdev, offset);
return (true);
}
```

predefined class variables

procedure called the first time a GraphTip is activated

specify size for graphic and mask

create graphics context for mask

procedure called when group is opened

provides access to time zone layer

called when DataTip event is triggered

translates object coordinates to window coordinates so can find polygon closest to cursor

finds polygon closest to cursor

get current time

assigns attribute value from closest polygon

adjust hour for half-hour time zones

create graphics context for clock

set clock face color according to local time and draw

convert hours and minutes to degrees

draw hour and minute hands

style and draw text for clock face

declare variables and convert tick position to degrees (repeated for 2,4,5,7,8,10, and 11)

draw ticks on clock face (repeated for 2,4,5,7,8,10, and 11)

sets offset for GraphTip and sets the rendered image and mask as its source