```
// ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
funcprot(0);
// ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

// Get body position from "cen" in EOD
// cjd: TREF time scale

function [pos_ECI]=getBodyPos(name, cjd)
  pos_ICRS=CL_eph_de405(name,cjd,"Earth");

  pos_ECI=CL_fr_convert("ICRS","ECI",cjd,pos_ICRS);
endfunction

// Azimuth and elevation of celestial body from location loc at times t

function [azim, elev]=sphPos(t, loc, cbody)
  pos_ECI=[];

  pos_ECI=getBodyPos(cbody,t);

  // Transform position of celestial body in ECF frame
  // (Earth Centered Fixed Frame)

  pos_ECF=CL_fr_convert("ECI","ECF",t,pos_ECI);

  // Transform elliptical coordinates of the position of celestial body
  // to cartesian coordinates

  loc_car=CL_co_ell2car(loc);

  // Get the transformation matrix to the local Topocentric North frame
  // The "topocentric North" local frame is defined as follows:
  // X: unit vector in the horizontal plane and towards norh
  // Y: unit vector in the horizontal plane and towards west
  // Z: unit vector in direction of the local vertical

  M_topoN=CL_fr_topoNMat(loc);

  // Transform position of celestial body to the local Topocentric North
  // frame in cartesian coordinates

  pos_topoN=M_topoN*(pos_ECF-loc_car);

  // Transform cartesian coordinates to spherical coordinates

  pos_topoN_sph=CL_co_car2sph(pos_topoN);

  // Get azimuth and elevation of celestial body

  azim=pos_topoN_sph(1);
  elev=pos_topoN_sph(2);

endfunction

// ####################################################################
```

```
// Main

// longitude/latitude/altitude of location

lon=11.42*%CL_deg2rad;
lat=48.15*%CL_deg2rad;
alt=531;

// scal=reference date (string, calendar)
scal=[];

desc_param=list(..
   CL_defParam("Longitude of location",lon,units=["rad","deg"],..
                valid='$x>=0 & $x <=360'), ..
   CL_defParam("Latitude of location",lat,units=["rad","deg"],..
                valid='$x>=-90 & $x <=90'), ..
   CL_defParam("Altitude of location",alt,units=["m"],..
                valid='$x>=-100 & $x <=3000'), ..
   CL_defParam("Reference time (calendar format, TREF)",scal,typ="cal") ..
);

[lon,lat,alt,scal]=CL_inputParam(desc_param);

loc=[lon;lat;alt];

t0=CL_dat_cal2cjd(CL_dat_str2cal(scal));

// ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

cal = CL_dat_cjd2cal(t0);

months=["Jan","Feb","Mär","Apr","Mai","Jun","Jul","Aug","Sep",..
          "Okt","Nov","Dez"];
str_day=sprintf("%d %s %d", cal(3), months(cal(2)), cal(1));
str_hour=sprintf("%02d:%02d", cal(4), cal(5));
dateText=str_day+" "+str_hour+" (TREF)";

// ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

function [razi]=cvAzi(pazi)
    razi=360-pazi*%CL_rad2deg;
endfunction

function [rele]=cvEle(pele)
    rele=pele*%CL_rad2deg;
endfunction

// ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

cBodyNames=["Sun","Moon","Mercury","Venus","Mars",..
            "Jupiter","Saturn","Uranus","Neptune","Pluto"];

pNames=["Sonne    ","Mond     ","Merkur   ","Venus    ","Mars     ",..
        "Jupiter  ","Saturn   ","Uranus   ","Neptun   ","Pluto    "];

```

```
pData=[];

NbBodies=size(cBodyNames,"c");

for Ix=1:NbBodies
    [azim,elev]=sphPos(t0,loc,cBodyNames(Ix));

    pData(Ix,1:2)=[cvAzi(azim),cvEle(elev)];
end

mprintf('\n\n   Sichtbarkeit von Himmelskörpern:\n');

mprintf('   Zeit - %s\n',dateText);

mprintf('   Standort -\n\tLänge:  %7.2f°\n\tBreite: %7.2f°\n\tHöhe:\t  %d m  \n',..
          lon*%CL_rad2deg,lat*%CL_rad2deg,alt);

mprintf('   Azimuth - Nord: 0° -> Ost: 90°\n\n');

mprintf('   Übern Horizont:\n');

for Ix=1:NbBodies

    if (pData(Ix,2)>0) then
        mprintf('   %s- Azimuth:  %11.6f°  |  Elevation:  %11.6f° \n',..
                pNames(Ix),pData(Ix,1),pData(Ix,2));
    end
end

mprintf('\n');
mprintf('   Unterm Horizont:\n');

for Ix=1:NbBodies

    if (pData(Ix,2)<=0) then
        mprintf('   %s- Azimuth:  %11.6f°  |  Elevation:  %11.6f° \n',..
                pNames(Ix),pData(Ix,1),pData(Ix,2));
    end
end

// ######################################################################

// name of body (capitalized)
function [name]=body_name(body)
  if (part(body,1:1) == 'm')
    name = "Moon";
  else
    name = "Sun";
  end
endfunction

// ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

// thickness after plot
```

```
function Thickness(th)
  e=gce();
  e.children.thickness = th;
endfunction

// ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

// xtring interface
function String(x, y, text, col, sz, style)
  xstring(x,y,text);
  e=gce();
  e.font_foreground = col;
  e.font_size = sz;
  e.font_style = style;
endfunction

// ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

// marks for sun and moon
function plot_mark(name, azi, ele, col, fg, sz)
  plot(azi,ele,"o");
  e=gce();
  e.children.mark_background=col;
  e.children.mark_foreground=fg;
  e.children.mark_size=sz;
  String(azi,ele," "+name, col,3,5);
endfunction

// ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

f=scf();
f.visible="off";
f.immediate_drawing="off";
a=gca();

col1 = color("orange"); // Sun
col2 = color("steelblue"); // Moon
col3 = color("darkseagreen"); // meridians
col4 = color("forestgreen"); // meridians

dbElev=80;
db = [0,-dbElev;360,dbElev]; // data bounds

// ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

col1=color("gold"); // Sonne
col2=color("magenta"); // Mond
col3=color("aquamarine"); // Merkur
col4=color("limegreen"); // Venus
col5=color("orangered"); // Mars
col6=color("darkgoldenrod"); // Jupiter
col7=color("royalblue"); // Saturn
col8=color("salmon"); // Uranus
col9=color("mediumorchid"); // Neptun
col10=color("peachpuff4"); // Pluto
```

```
plotNames=["SO","MO","Me","Ve","Ma","Ju","Sa","Ur","Ne","Pl"];

sizes=[15, 14, 9, 12, 10, 11, 8, 7, 6, 5];
//     SO  MO  Me Ve  Ma  Ju Sa  Ur Ne Pl

cols=[col1,col2,col3,col4,col5,col6,col7,col8,col9,col10];
//     SO   MO   Me   Ve   Ma   Ju   Sa   Ur    Ne   Pl

for Ix=1:NbBodies
    plot_mark(plotNames(Ix),pData(Ix,1),pData(Ix,2),cols(Ix),1,sizes(Ix));
end

// ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

// various settings

a.x_ticks = tlist("ticks", 0:60:360, string(0:60:360));
a.y_ticks = tlist("ticks", -dbElev:10:dbElev, string(-dbElev:10:dbElev));
a.x_label.text = "Azimuth [°]";
a.y_label.text = "Elevation [°]";


months = ["Jan", "Feb", "Mär", "Apr", "Mai", "Jun", "Jul", "Aug", "Sep", "Okt", "Nov", "

str_day = sprintf("%d %s %d", cal(3), months(cal(2)), cal(1));
str_hour = sprintf("%02d:%02d", cal(4), cal(5));

a.title.text = "Sonne, Mond und Planeten
 - " + str_day + " " + str_hour + " (TREF)";
a.data_bounds = db;

CL_g_stdaxes(a);
a.grid = [0,0];
xgrid(color("grey"));

CL_g_legend(a, str=['Sonne';'Mond';'Merkur';'Venus';'Mars';'Jupiter';..
                    'Saturn';'Uranus';'Neptun';'Pluto']);

f.immediate_drawing="on";
f.visible="on";
```