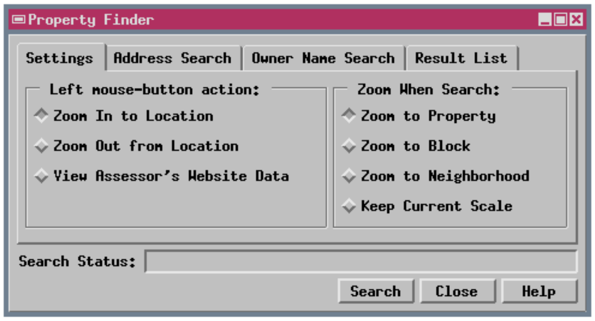
Property Finder Tool Script

When you are preparing your geospatial data for distribution as an atlas, you can use SML Tool Scripts to provide the atlas with custom data-exploration tools that are powerful yet simple to use, even for those who have little or no experience with the TNT products. One example is the Property Finder Tool Script (excerpted on the other side of this plate), developed in conjunction with the Property Viewer atlas, a city-wide electronic atlas of Lincoln, Nebraska that features a high-resolution color orthoimage, property ownership information, and many other thematic layers (see the color plate entitled *Property Viewer Atlas for Lincoln, NE*). You do not need to be a professional programmer to develop such scripts; the Property Finder Tool Script was written by the author of this plate, a technical writer with a geology background and modest geospatial scripting experience.



Property Finder

Settings | Address Search | Owner Name Search | Result List

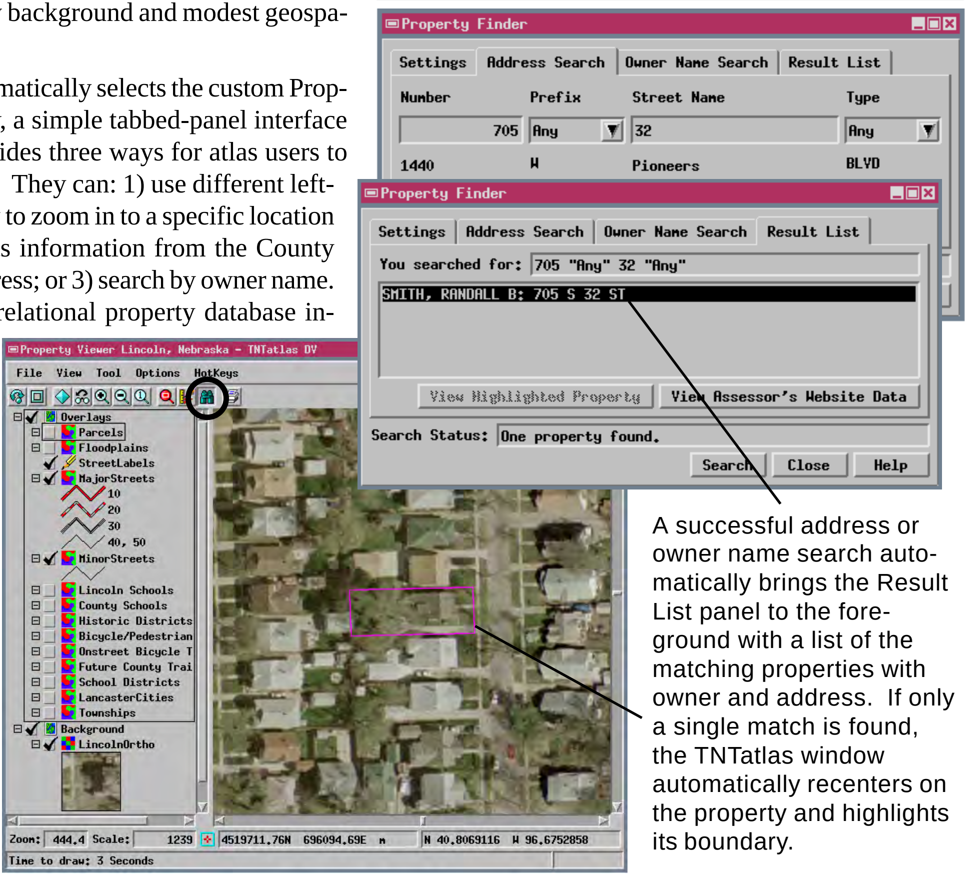
Left mouse-button action:
◇ Zoom In to Location
◇ Zoom Out from Location
◇ View Assessor's Website Data

Zoom When Search:
◇ Zoom to Property
◇ Zoom to Block
◇ Zoom to Neighborhood
◇ Keep Current Scale

Search Status:

Upon opening, the Property Viewer atlas automatically selects the custom Property Finder tool and opens its dialog window, a simple tabbed-panel interface designed in XML. The Property Finder provides three ways for atlas users to locate particular property parcels in the atlas. They can: 1) use different left-mouse button actions in the TNTatlas window to zoom in to a specific location in the view and select a property to show its information from the County Assessor's website; 2) search by property address; or 3) search by owner name. The latter two methods utilize the detailed relational property database included with the atlas. The Tool Script uses the address or owner name provided by the atlas user to construct a selection query on this database. The atlas user needs no prior knowledge of the database content and structure and no knowledge of TNT query syntax or structure.

The property database and the search procedures in this Tool Script have been optimized to speed searches through the 100,000 property records in the database. As a result, search and display take only a few seconds even when the atlas is run directly from the DVD. Address searches require an exact match between the address number and street name entered on the Address Search panel and the corresponding fields in the property database. The atlas user can optionally select a prefix (N, S, ...) and a street type (RD, ST, ...) from menus to narrow the search. Owner name searches are less restrictive, despite the fact that a single field in the property database contains all the ownership information for each property (last, first, and middle names for one or more owners). The last name entered on the Owner Name Search panel must match a last name in the database field exactly, but the entered first name and middle initial combined need only match the *beginning* of a first name / middle name combination in this field. The atlas user can therefore enter either a first name plus middle initial (narrowest search), a first name only, or a partial first name (even just a first initial) for broader searches. For jointly owned properties, a matching first name / middle name can be found regardless of what position it occupies in the composite owner name field in the database.



Property Finder

Settings | Address Search | Owner Name Search | Result List

Number	Prefix	Street Name	Type
705	Any ▼	32	Any ▼
1440	W	Pioneers	BLVD

Property Finder

Settings | Address Search | Owner Name Search | Result List

You searched for: 705 "Any" 32 "Any"

SMITH, RANDALL B: 705 S 32 ST

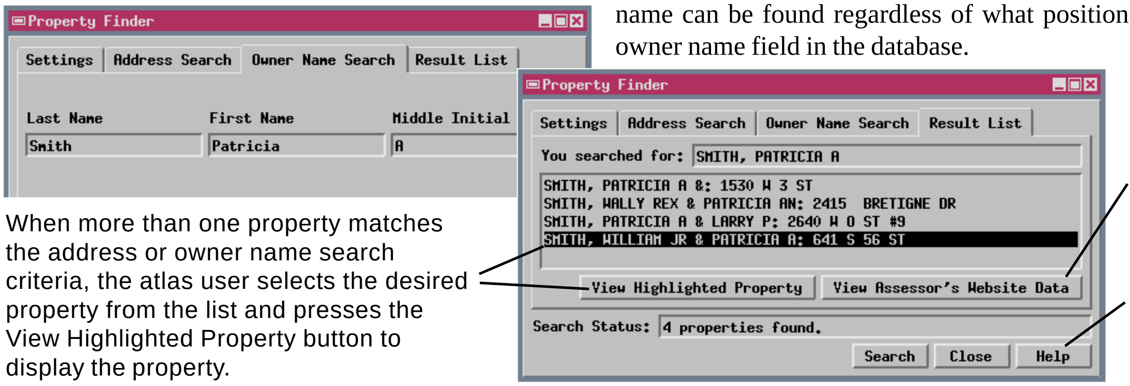
Search Status: One property found.

Property Viewer Lincoln, Nebraska - TNTatlas DV

Zoom: 444.4 Scale: 1239 4519711.76N 696094.69E m N 40.8069116 W 96.6752858

Time to draw: 3 Seconds

A successful address or owner name search automatically brings the Result List panel to the foreground with a list of the matching properties with owner and address. If only a single match is found, the TNTatlas window automatically recenters on the property and highlights its boundary.



Property Finder

Settings | Address Search | Owner Name Search | Result List

Last Name	First Name	Middle Initial
Smith	Patricia	A

Property Finder

Settings | Address Search | Owner Name Search | Result List

You searched for: SMITH, PATRICIA A

```
SMITH, PATRICIA A &: 1530 W 3 ST
SMITH, WALLY REX & PATRICIA AN: 2415  BRETIGNE DR
SMITH, PATRICIA A & LARRY P: 2640 W O ST #9
SMITH, WILLIAM JR & PATRICIA A: 641 S 56 ST
```

Search Status: 4 properties found.

When more than one property matches the address or owner name search criteria, the atlas user selects the desired property from the list and presses the View Highlighted Property button to display the property.

Assessor's information can also be accessed for properties found by address or owner name search.

The Help button launches the default Browser and shows an illustrated HTML file with instructions for using the Property Finder.

Macro and Tool Scripts can be created using SML in any TNTmips process that uses a View window (Options / Scripts from the View window menu bar). These scripts are then available from an icon, which you select or design, on the toolbar. Sample scripts have been prepared to illustrate how you might use these features to assist with specific tasks you perform on a regular basis. If possible, the full script is printed below for your quick perusal. When a script is too long to fit on one page, key sections are reproduced below. The sample script shown below can be downloaded from the SML script exchange at www.microimages.com/freestuf/scripts.htm.

Script Excerpt for Property Finder (PropFinder.sml)

```
proc OnSearch ()
   {
```

called when Search button on Property Search dialog is pressed

```
   settings = dlgwin.GetValues();
   local numeric possibleRecList[1];
   local numeric numRecords, record;
```

get control settings (values) from dialog and store in GUI_FORMDATA

```
   if (searchmode == 1)
      {
      ...
      }
```

address search; components of address are in separate fields in the database (script code omitted here)

```
   else if (searchmode == 0)
      {
```

owner name search; entire name(s) is in single field in database

```
      local string lastname$, firstname$, middle$;

      local string firstmid$;
```

concatenation of first name and middle initial

```
      local string fullname$;
      local numeric firstmidlength;

      local numeric numTokens;
      local numeric j;
      local numeric name_in;
```

number of tokens in owner name (delimited by space)

```
      local class STRINGLIST tokenList;
```

stringlist containing tokens from name

read name values from formdata and set all to upper case

```
      lastname$ = toupper$(settings.GetValueStr("lastnamefld") );
      firstname$ = toupper$(settings.GetValueStr("firstnamefld") );
      middle$ = toupper$(settings.GetValueStr("midintfld") );
```

concatenate middle initial with first name

```
      if (middle$ <> "") then
         firstmid$ = sprintf("%s %s", firstname$, middle$);
      else
         firstmid$ = firstname$;

      firstmidlength = strlen(firstmid$);
```

get length of string to use later in search

concatenate lastname and firstmid to use to display search text

```
      fullname$ = sprintf("%s, %s", lastname$, firstmid$);
```

don't search if no owner name is entered in dialog

```
      if (fullname$ == ", ")
         {
         statusText.SetValue("Please enter an owner name.", 0);
         }
```

if name is same as previous, abort search and show previous results

```
      else if ( (fullname$ == prevname$) && (prevsearchmode == 0) )
         if (count == 0) then
            OnFoundNone();
         else
            OnFound(count);

      else
         {
```

otherwise do new search

```
         prevsearchmode = 0;
```

NOTE: this script requires TNTmips or TNTatlas V7.0 DV dated 14 April 2004 or later.

record name string for comparison on next search

```
         prevname$ = fullname$;
         searchText.SetValue(fullname$, 0);

         OnNewSearch();
```

search through parcel polygons

```
         count = 0;
         ResizeArrayClear(matchRecList, 0);
```

reinitialize array holding numbers of matching records

```
         ResizeArrayClear(propsArray, 0);
```

reinitialize array holding found polygon numbers

Narrow down the search from 99,000 records to just the records in which the OWNER string begins with the requested last name. Uses a function that takes advantage of an index on the selected field to speed the search

```
         numRecords = TableKeyFieldLookupList(parcelVector.poly.OwnerName,
               "OWNER", lastname$, possibleRecList, "StartsWith" );
```

Now search the returned records for those with a match to the requested name

```
         for i = 1 to numRecords
            {
            local numeric name_in = 0;
            View.SetStatusBar(i, numRecords);
            record = possibleRecList[i];
```

update View window's status bar

get owner name string from database

```
            local string name$ =
                  parcelVector.poly.OwnerName[@record].OWNER$;
            local class STRING teststr$;
```

check for exact match with last name (not just starts with)

```
            if ( GetToken(name$, ",", 1) == lastname$)
               {
```

number of tokens in owner name with "," and "&" as delimiters

```
               numTokens = NumberTokens(name$, ",&");
```

loop through tokens following last name

```
               for j = 2 to numTokens
                  {
                  teststr$ = GetToken(name$, ",&", j);
                  teststr$.remove(0, 0);
```

remove leading space from test string

```
                  if ( left$( teststr$, firstmidlength) == firstmid$) then
                     name_in = 1;
                  }
               }
```

token begins with name string; set flag to get polygon number

```
            if (name_in == 1)
               {
               count = count + 1;
               ResizeArrayPreserve(matchRecList, count + 1);
```

increase array size for polygon found

```
               matchRecList[count] = record;
               }
            }
```

add polygon number to array

```
         GetPolygons();
```

call procedure to get polygons corresponding to matching records

```
         }
      }
   }
```

end else do new search

end else if (searchmode == 0)

end OnSearch