GUI apps with R & tcltk

An example based on functionality in ZoopSynth

Travis Hinkelman (ESA) • IEP Data Science PWT • 2024-12-12

GUI Packages

- Shiny
  - Web framework
  - Modern, big ecosystem

- RGtk2
  - Package removed from CRAN in 2021

- qtbase
  - GitHub repo last updated 5 years ago
  - Package removed from CRAN in 2020

- tcltk
  - Included in base R distribution
  - Dated, limited widgets

- tcltk2
  - Additional commands and widgets

- tickle
  - Opinionated abstraction over tcltk

GUI Packages

- Shiny
  - Web framework
  - Modern, big ecosystem

- RGtk2
  - Package removed from CRAN in 2021

- qtbase
  - GitHub repo last updated 5 years ago
  - Package removed from CRAN in 2020

- tcltk
  - Included in base R distribution
  - Dated, limited widgets

- tcltk2
  - Additional commands and widgets

- tickle
  - Opinionated abstraction over tcltk

GUI Packages

- Shiny
  - Web framework
  - Modern, big ecosystem

- RGtk2
  - Package removed from CRAN in 2021

- qtbase
  - GitHub repo last updated 5 years ago
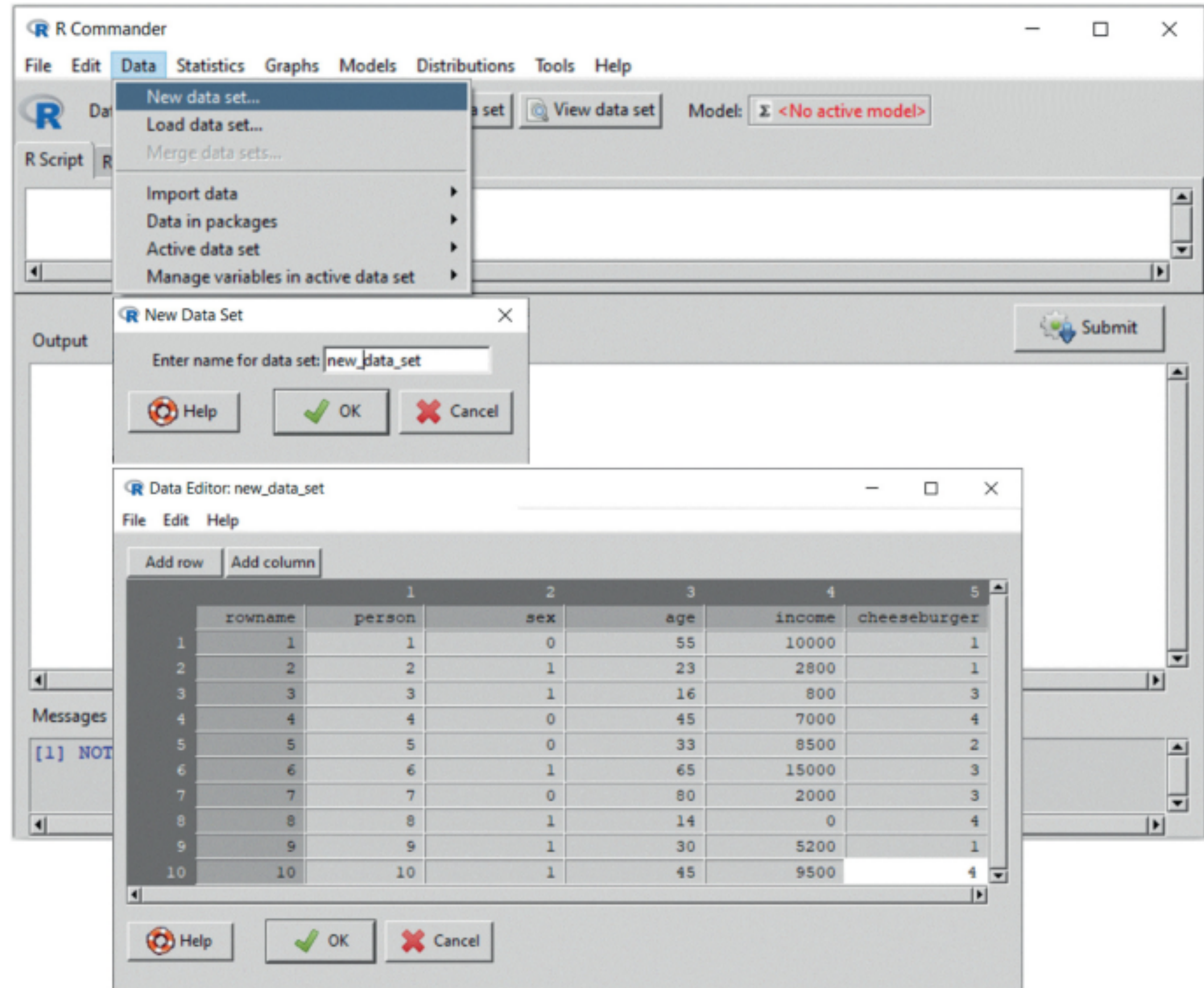 - Package removed from CRAN in 2020

Tcl/Tk: tool command language/toolkit (pronounced tickle-teak)

Many languages have bindings to Tcl/Tk, including Python, Ruby, and Perl

- tcltk
  - Included in base R distribution
  - Dated, limited widgets

- tcltk2
  - Additional commands and widgets

- tickle
  - Opinionated abstraction over tcltk

R Commander

- Rcmdr package

- GUI for R statistics and graphics

- Built with tcltk package

Demo

R Packages & Variables

```r
library(tcltk)
library(zooper)
library(lubridate)
library(dplyr)
library(ggplot2)

source_names = c("Environmental Monitoring Program (EMP)", "Fall Midwater Trawl (FMWT)",
                 "Summer Townet Survey (STN)", "20mm Survey (20mm)",
                 "Fish Restoration Program (FRP)", "Directed Outflow Project (DOP)")
source_codes = c("EMP", "FMWT", "STN", "20mm", "FRP", "DOP")
source_colors <- setNames(RColorBrewer::brewer.pal(6, "Set2"), source_codes)
size_codes = c("Micro", "Meso", "Macro")
zoop_data = NULL
```

R Packages & Variables

```r
library(tcltk)
library(zooper)
library(lubridate)
library(dplyr)
library(ggplot2)

source_names = c("Environmental Monitoring Program (EMP)", "Fall Midwater Trawl (FMWT)",
    "Summer Townet Survey (STN)", "20mm Survey (20mm)",
    "Fish Restoration Program (FRP)", "Directed Outflow Project (DOP)")
source_codes = c("EMP", "FMWT", "STN", "20mm", "FRP", "DOP")
source_colors <- setNames(RColorBrewer::brewer.pal(6, "Set2"), source_codes)
size_codes = c("Micro", "Meso", "Macro")
zoop_data = NULL
```

tcl Variables

tclVar() creates binding between R and tcl variables

```
datatype = tclVar("Community")
```

```
sources = tclVar()
tclvalue(sources) = source_names
```

```
size_classes = tclVar()
tclvalue(size_classes) = size_codes
```

```
min_yr = tclVar(1972)
max_yr = tclVar(2020)
```

```
months = tclVar()
tclvalue(months) = month.name
```

```
taxa = tclVar()
tclvalue(taxa) = completeTaxaList
```

tcl Variables

Initial Values

```
datatype = tclVar("Community")

sources = tclVar()
tclvalue(sources) = source_names

size_classes = tclVar()
tclvalue(size_classes) = size_codes

min_yr = tclVar(1972)
max_yr = tclVar(2020)

months = tclVar()
tclvalue(months) = month.name

taxa = tclVar()
tclvalue(taxa) = completeTaxaList
```

tcl Variables

Choices

```
datatype = tclVar("Community")
```

```
sources = tclVar()
tclvalue(sources) = source_names
```

```
size_classes = tclVar()
tclvalue(size_classes) = size_codes
```

```
min_yr = tclVar(1972)
max_yr = tclVar(2020)
```

```
months = tclVar()
tclvalue(months) = month.name
```

```
taxa = tclVar()
tclvalue(taxa) = completeTaxaList
```

Named Frames & Widgets

```r
base <- tktoplevel()
tkwm.title(base, "ZoopSynth")
mainframe = ttkframe(base, padding = c(10, 10, 10, 10))

sources_lb = tklistbox(mainframe, listvariable = sources, selectmode = "multiple",
                       exportselection = FALSE, height = 6)

sizes_lb = tklistbox(mainframe, listvariable = size_classes, selectmode = "multiple",
                     exportselection = FALSE, height = 3)

months_lb = tklistbox(mainframe, listvariable = months, selectmode = "multiple",
                      exportselection = FALSE, height = 6)

taxa_lb = tklistbox(mainframe, listvariable = taxa, selectmode = "multiple",
                    exportselection = FALSE, height = 6, state = "disabled")
```

Named Frames & Widgets

```
base <- tktoplevel()
tkwm.title(base, "ZoopSynth")
mainframe = ttkframe(base, padding = c(10, 10, 10, 10))

sources_lb = tklistbox(mainframe, listvariable = sources, selectmode = "multiple",
                       exportselection = FALSE, height = 6)

sizes_lb = tklistbox(mainframe, listvariable = size_classes, selectmode = "multiple",
                     exportselection = FALSE, height = 3)

months_lb = tklistbox(mainframe, listvariable = months, selectmode = "multiple",
                      exportselection = FALSE, height = 6)

taxa_lb = tklistbox(mainframe, listvariable = taxa, selectmode = "multiple",
                    exportselection = FALSE, height = 6, state = "disabled")
```
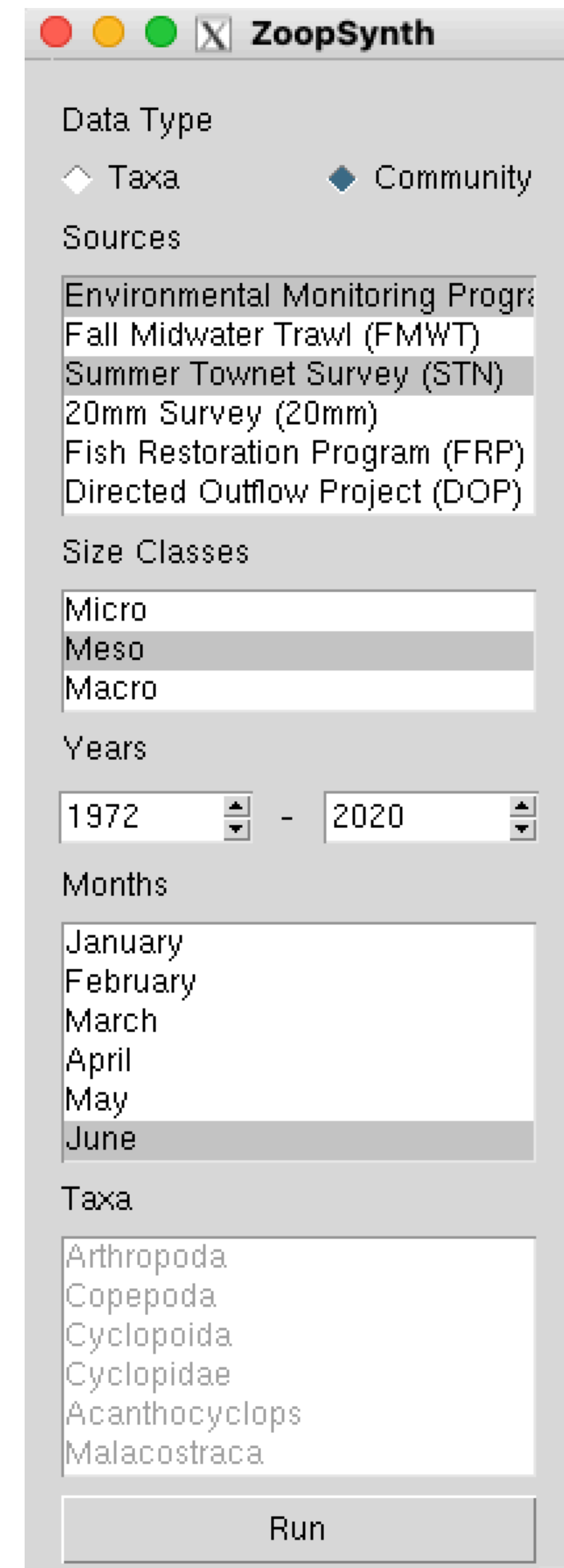
Named Frames & Widgets


```
base <- tktoplevel()
tkwm.title(base, "ZoopSynth")
mainframe = ttkframe(base, padding = c(10, 10, 10, 10))

sources_lb = tklistbox(mainframe, listvariable = sources, selectmode = "multiple",
                       exportselection = FALSE, height = 6)

sizes_lb = tklistbox(mainframe, listvariable = size_classes, selectmode = "multiple",
                     exportselection = FALSE, height = 3)

months_lb = tklistbox(mainframe, listvariable = months, selectmode = "multiple",
                      exportselection = FALSE, height = 6)

taxa_lb = tklistbox(mainframe, listvariable = taxa, selectmode = "multiple",
                    exportselection = FALSE, height = 6, state = "disabled")
```

Named Frames & Widgets

```r
base <- tktoplevel()
tkwm.title(base, "ZoopSynth")
mainframe = ttkframe(base, padding = c(10, 10, 10, 10))

sources_lb = tklistbox(mainframe, listvariable = sources, selectmode = "multiple",
                       exportselection = FALSE, height = 6)

sizes_lb = tklistbox(mainframe, listvariable = size_classes, selectmode = "multiple",
                     exportselection = FALSE, height = 3)

months_lb = tklistbox(mainframe, listvariable = months, selectmode = "multiple",
                      exportselection = FALSE, height = 6)

taxa_lb = tklistbox(mainframe, listvariable = taxa, selectmode = "multiple",
                    exportselection = FALSE, height = 6, state = "disabled")
```
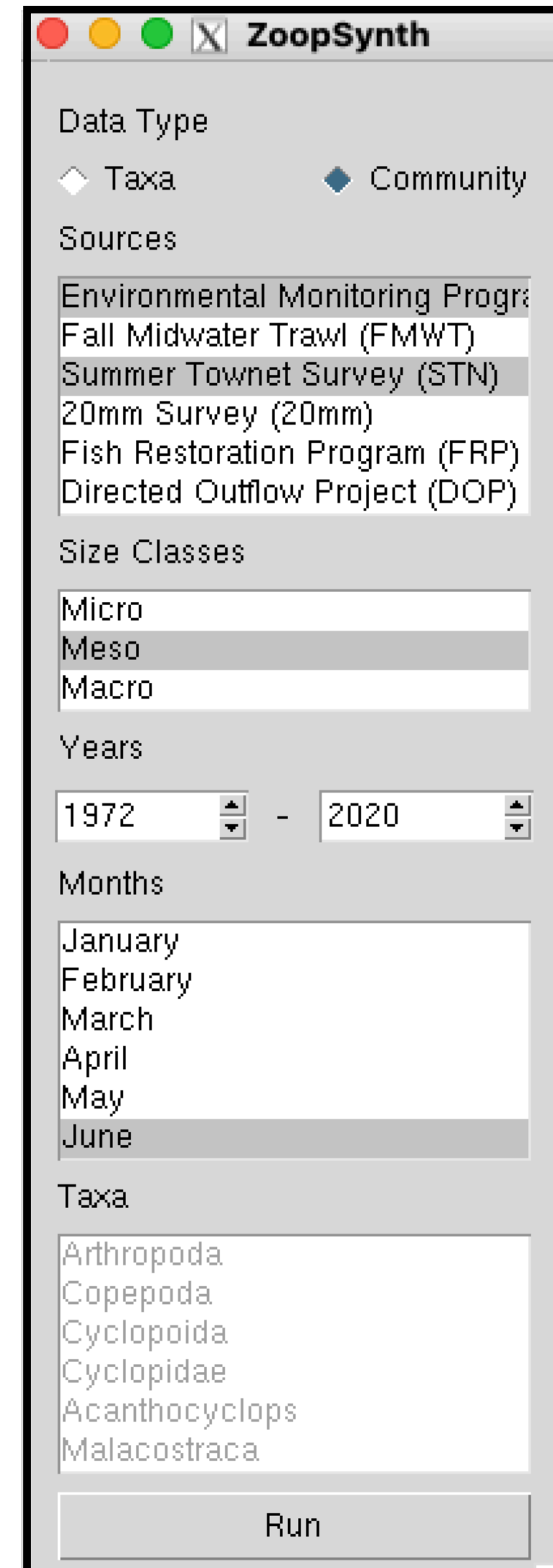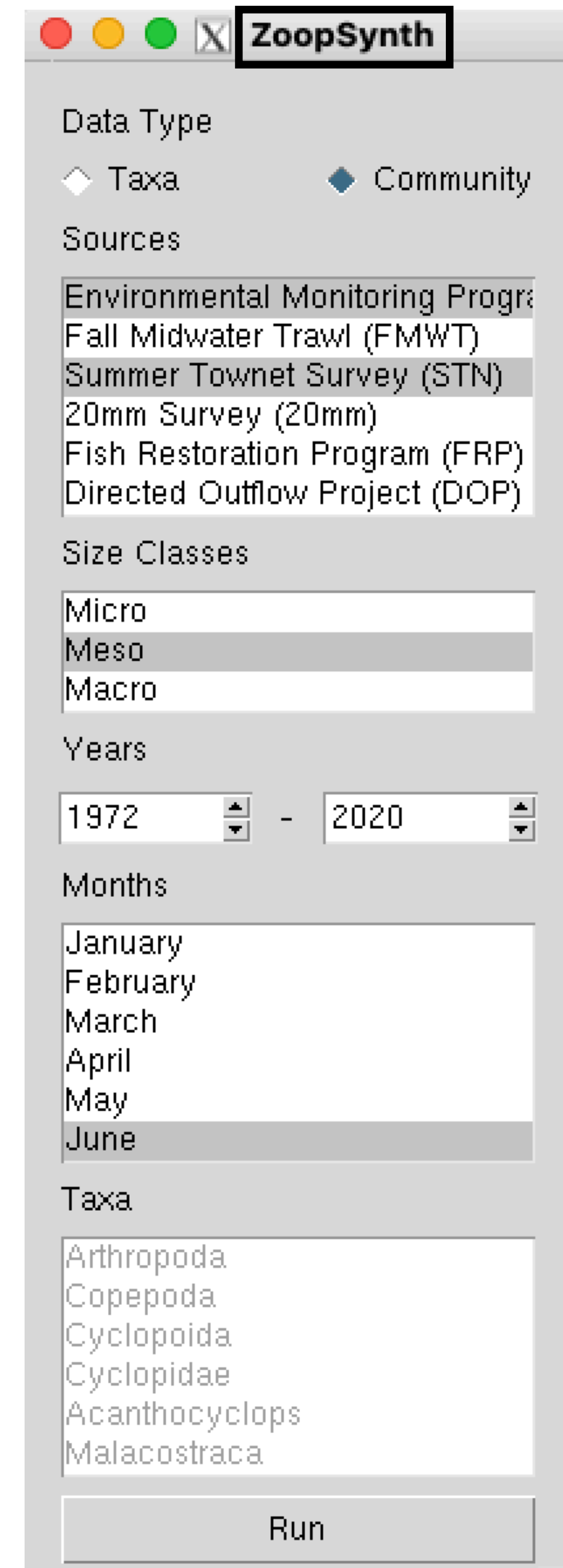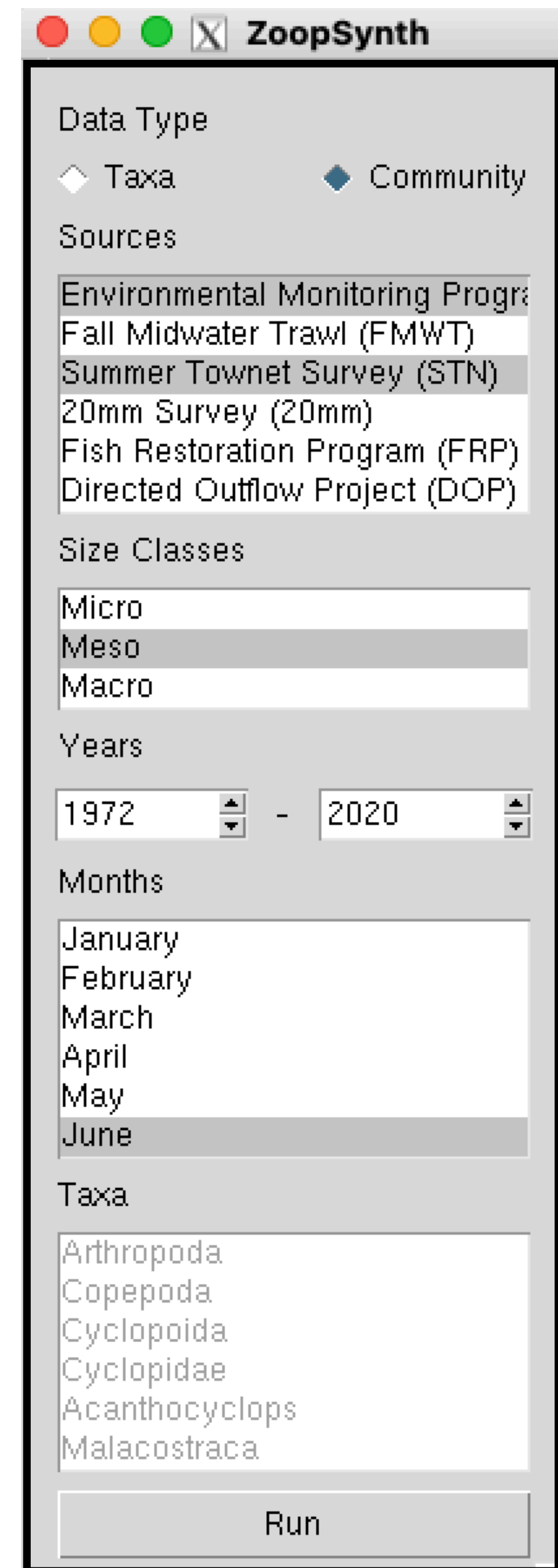
Named Frames & Widgets

```
base <- tktoplevel()
tkwm.title(base, "ZoopSynth")
mainframe = ttkframe(base, padding = c(10, 10, 10, 10))

sources_lb = tklistbox(mainframe, listvariable = sources, selectmode = "multiple",
                       exportselection = FALSE, height = 6)

sizes_lb = tklistbox(mainframe, listvariable = size_classes, selectmode = "multiple",
                     exportselection = FALSE, height = 3)

months_lb = tklistbox(mainframe, listvariable = months, selectmode = "multiple",
                      exportselection = FALSE, height = 6)

taxa_lb = tklistbox(mainframe, listvariable = taxa, selectmode = "multiple",
                    exportselection = FALSE, height = 6, state = "disabled")
```
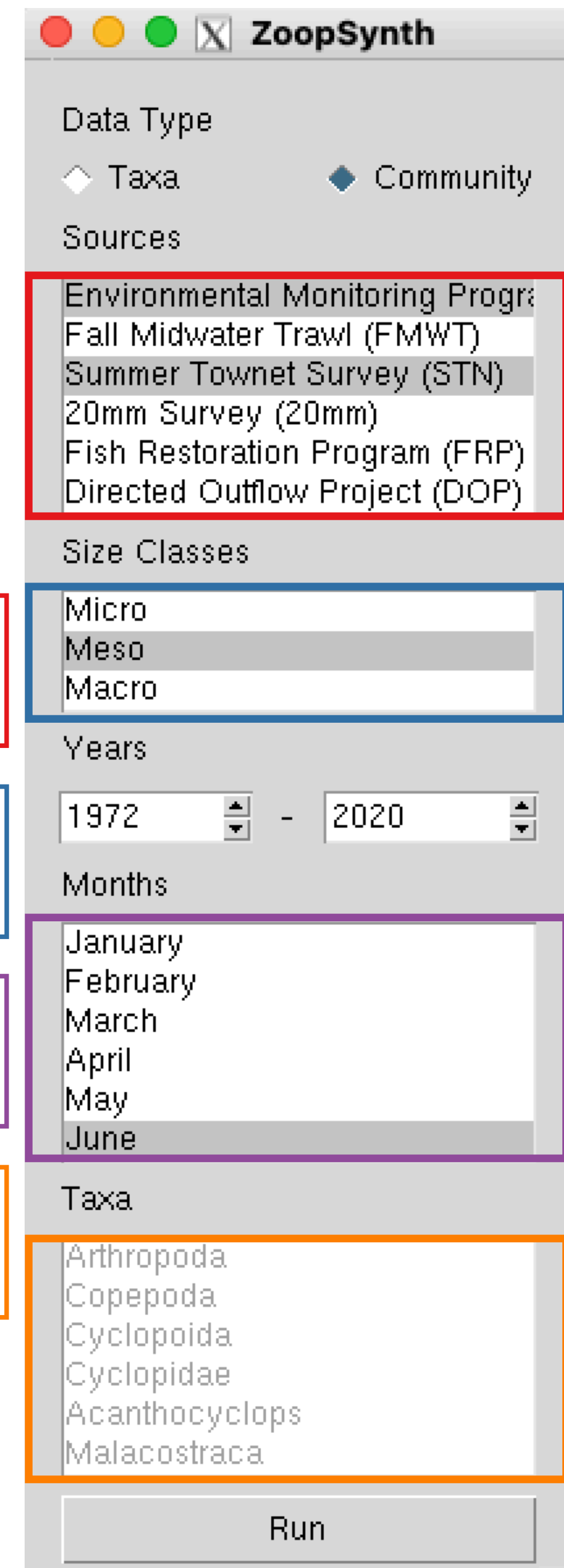
Named Frames & Widgets


```r
base <- tktoplevel()
tkwm.title(base, "ZoopSynth")
mainframe = ttkframe(base, padding = c(10, 10, 10, 10))

sources_lb = tklistbox(mainframe, listvariable = sources, selectmode = "multiple",
                       exportselection = FALSE, height = 6)

sizes_lb = tklistbox(mainframe, listvariable = size_classes, selectmode = "multiple",
                     exportselection = FALSE, height = 3)

months_lb = tklistbox(mainframe, listvariable = months, selectmode = "multiple",
                      exportselection = FALSE, height = 6)

taxa_lb = tklistbox(mainframe, listvariable = taxa, selectmode = "multiple",
                    exportselection = FALSE, height = 6, state = "disabled")
```

listvariable needs to be a tcl variable.

Named Frames & Widgets


```
base <- tktoplevel()
tkwm.title(base, "ZoopSynth")
mainframe = ttkframe(base, padding = c(10, 10, 10, 10))

sources_lb = tklistbox(mainframe, listvariable = sources, selectmode = "multiple",
                exportselection = FALSE, height = 6)

sizes_lb = tklistbox(mainframe, listvariable = size_classes, selectmode = "multiple",
                exportselection = FALSE, height = 3)

months_lb = tklistbox(mainframe, listvariable = months, selectmode = "multiple",
                exportselection = FALSE, height = 6)

taxa_lb = tklistbox(mainframe, listvariable = taxa, selectmode = "multiple",
                exportselection = FALSE, height = 6, state = "disabled")
```
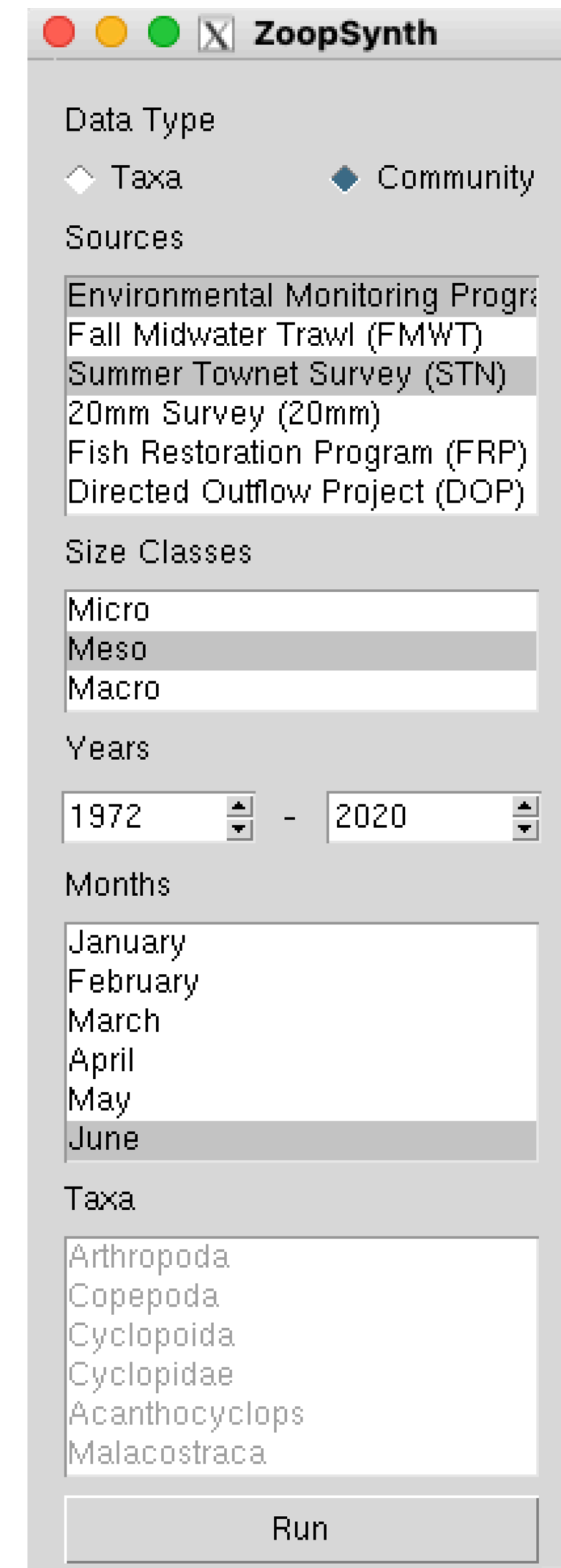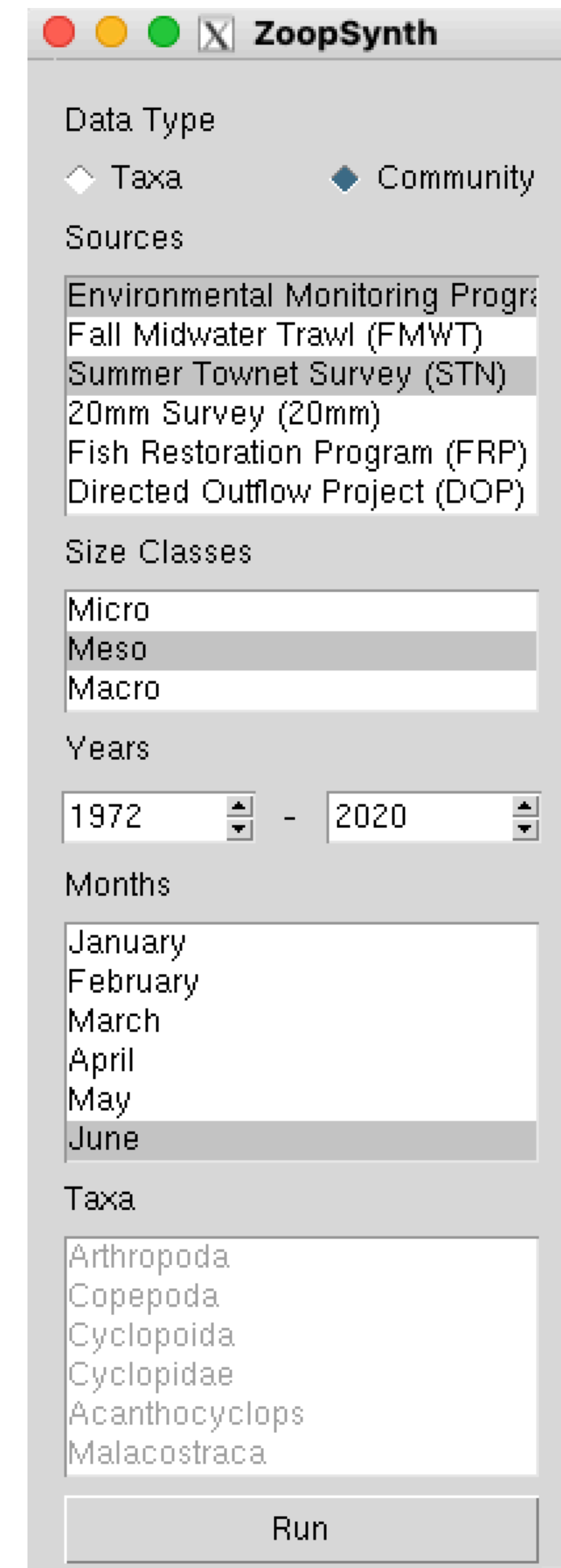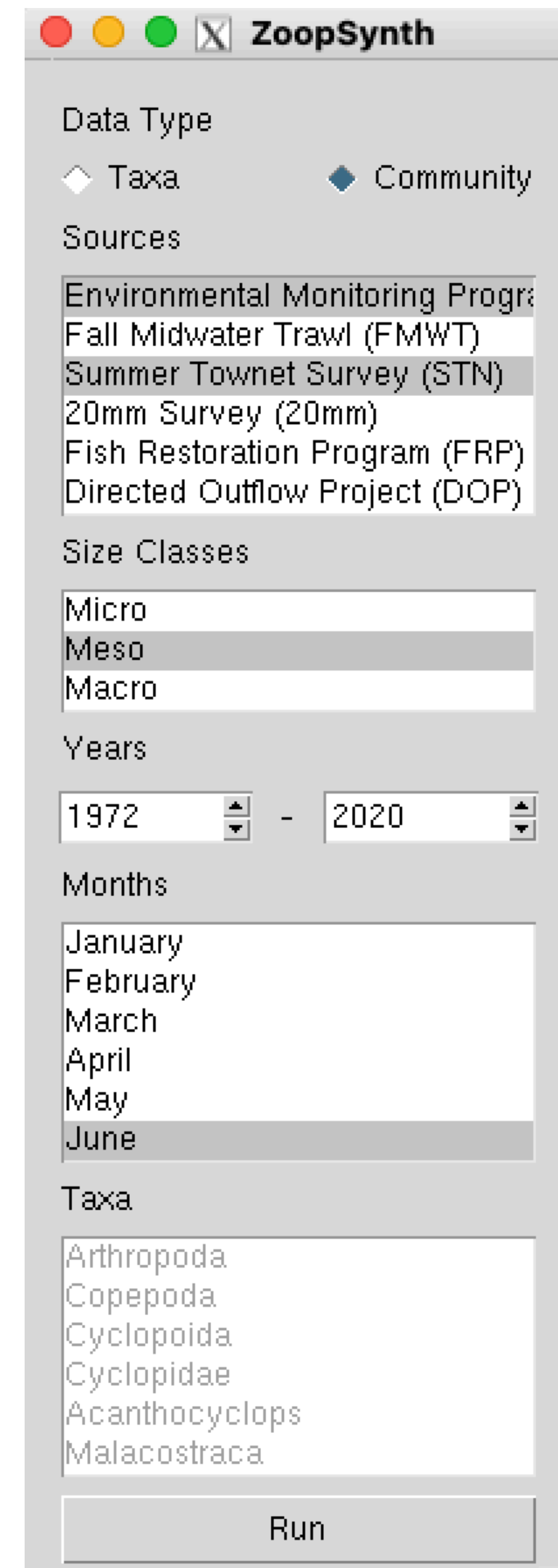
When exportselection = TRUE, clicking in one listbox removes selections for other listboxes.

Initial Listbox Selections

```r
# need to loop through selections when not consecutive
for (i in c(0, 2)) tkselection.set(sources_lb, i)
# only select one size class initially
tkselection.set(sizes_lb, 1)
# if consecutive, can specify first and last index to select multiple
tkselection.set(months_lb, 5, 7)
# indexing is 0-based so this selects Jun-Aug
```

Layout

- Three geometry managers

  - Grid

  - Pack

  - Place

- Grid is recommend over pack.

- Place is for low-level control.

Radio Buttons


```
tkgrid(ttklabel(mainframe, text = "Data Type"),
       row = 0, column = 0, sticky = "w", padx = 5, pady = 5)

tkgrid(ttkradiobutton(mainframe, text = "Taxa", value = "Taxa", variable = datatype,
                      command = function(...) tkconfigure(taxa_lb, state = "normal")),
       row = 1, column = 0, sticky = "w", padx = 5)

tkgrid(ttkradiobutton(mainframe, text = "Community", value = "Community", variable = datatype,
                      command = function(...) tkconfigure(taxa_lb, state = "disabled")),
       row = 1, column = 2, sticky = "w", padx = 5)
```

Radio Buttons

```
tkgrid(ttklabel(mainframe, text = "Data Type"),
       row = 0, column = 0, sticky = "w", padx = 5, pady = 5)

tkgrid(ttkradiobutton(mainframe, text = "Taxa", value = "Taxa", variable = datatype,
                      command = function(...) tkconfigure(taxa_lb, state = "normal")),
       row = 1, column = 0, sticky = "w", padx = 5)

tkgrid(ttkradiobutton(mainframe, text = "Community", value = "Community", variable = datatype,
                      command = function(...) tkconfigure(taxa_lb, state = "disabled")),
       row = 1, column = 2, sticky = "w", padx = 5)
```

Radio Buttons

```
tkgrid(ttklabel(mainframe, text = "Data Type"),
    row = 0, column = 0, sticky = "w", padx = 5, pady = 5)

tkgrid(ttkradiobutton(mainframe, text = "Taxa", value = "Taxa", variable = datatype,
                command = function(...) tkconfigure(taxa_lb, state = "normal")),
    row = 1, column = 0, sticky = "w", padx = 5)

tkgrid(ttkradiobutton(mainframe, text = "Community", value = "Community", variable = datatype,
                command = function(...) tkconfigure(taxa_lb, state = "disabled")),
    row = 1, column = 2, sticky = "w", padx = 5)
```

Spinbox

```
tkgrid(ttklabel(mainframe, text = "Years"),
    row = 6, column = 0, sticky = "we", padx = 5, pady = 5)
```

```
tkgrid(ttkspinbox(mainframe, textvariable = min_yr, from = 1972, to = 2020,
            increment = 1, width = 5),
        row = 7, column = 0, sticky = "we", padx = 5)
```

```
tkgrid(ttklabel(mainframe, text = "-"),
    row = 7, column = 1, sticky = "we", padx = 5, pady = 5)
```

```
tkgrid(ttkspinbox(mainframe, textvariable = max_yr, from = 1972, to = 2020,
            increment = 1, width = 5),
        row = 7, column = 2, sticky = "we", padx = 5)
```

Spinbox

```
tkgrid(ttklabel(mainframe, text = "Years"),
    row = 6, column = 0, sticky = "we", padx = 5, pady = 5)

tkgrid(ttkspinbox(mainframe, textvariable = min_yr, from = 1972, to = 2020,
            increment = 1, width = 5),
    row = 7, column = 0, sticky = "we", padx = 5)

tkgrid(ttklabel(mainframe, text = "-"),
    row = 7, column = 1, sticky = "we", padx = 5, pady = 5)

tkgrid(ttkspinbox(mainframe, textvariable = max_yr, from = 1972, to = 2020,
            increment = 1, width = 5),
    row = 7, column = 2, sticky = "we", padx = 5)
```

Spinbox

```
tkgrid(ttklabel(mainframe, text = "Years"),
    row = 6, column = 0, sticky = "we", padx = 5, pady = 5)

tkgrid(ttkspinbox(mainframe, textvariable = min_yr, from = 1972, to = 2020,
        increment = 1, width = 5),
    row = 7, column = 0, sticky = "we", padx = 5)

tkgrid(ttklabel(mainframe, text = "-"),
    row = 7, column = 1, sticky = "we", padx = 5, pady = 5)

tkgrid(ttkspinbox(mainframe, textvariable = max_yr, from = 1972, to = 2020,
        increment = 1, width = 5),
    row = 7, column = 2, sticky = "we", padx = 5)
```
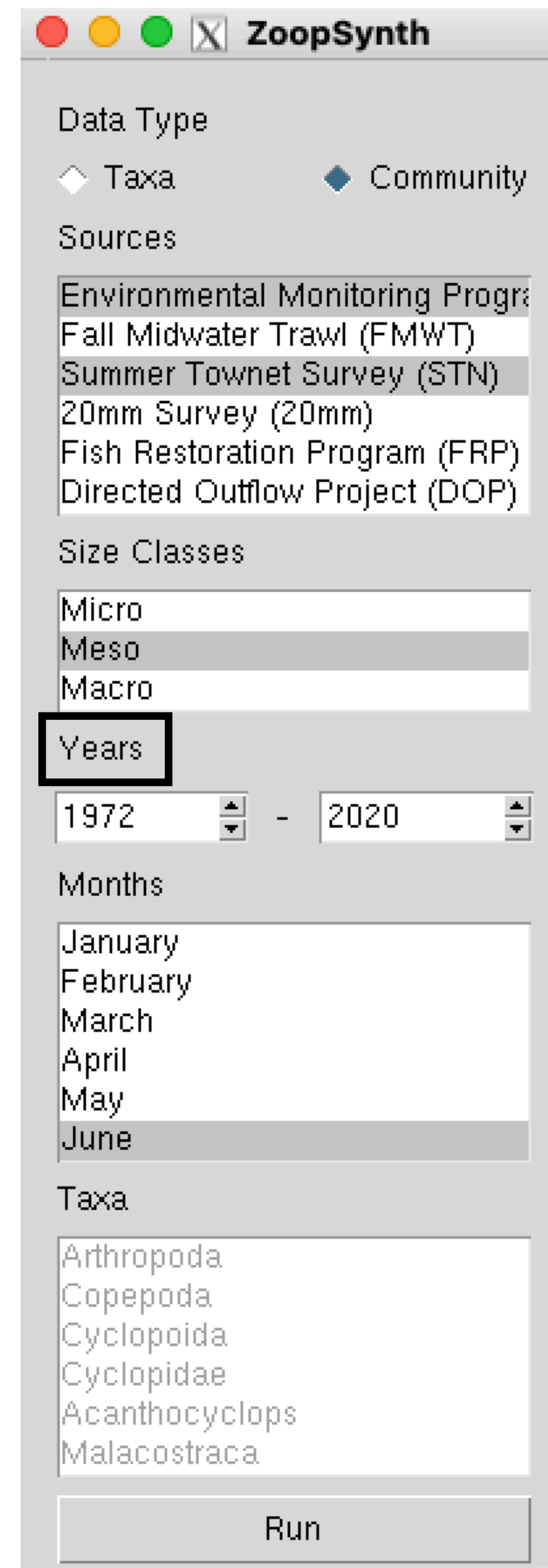
Spinbox


```
tkgrid(ttklabel(mainframe, text = "Years"),
    row = 6, column = 0, sticky = "we", padx = 5, pady = 5)


tkgrid(ttkspinbox(mainframe, textvariable = min_yr, from = 1972, to = 2020,
            increment = 1, width = 5),
    row = 7, column = 0, sticky = "we", padx = 5)


tkgrid(ttklabel(mainframe, text = "-"),
    row = 7, column = 1, sticky = "we", padx = 5, pady = 5)


tkgrid(ttkspinbox(mainframe, textvariable = max_yr, from = 1972, to = 2020,
            increment = 1, width = 5),
    row = 7, column = 2, sticky = "we", padx = 5)
```
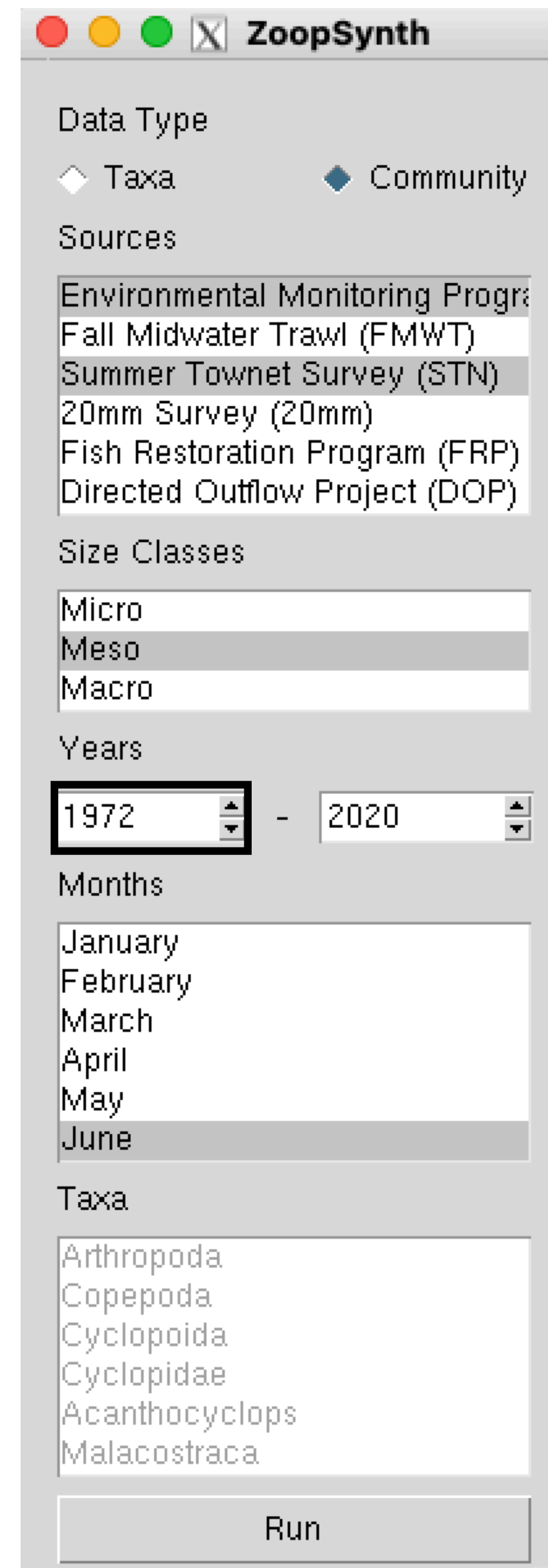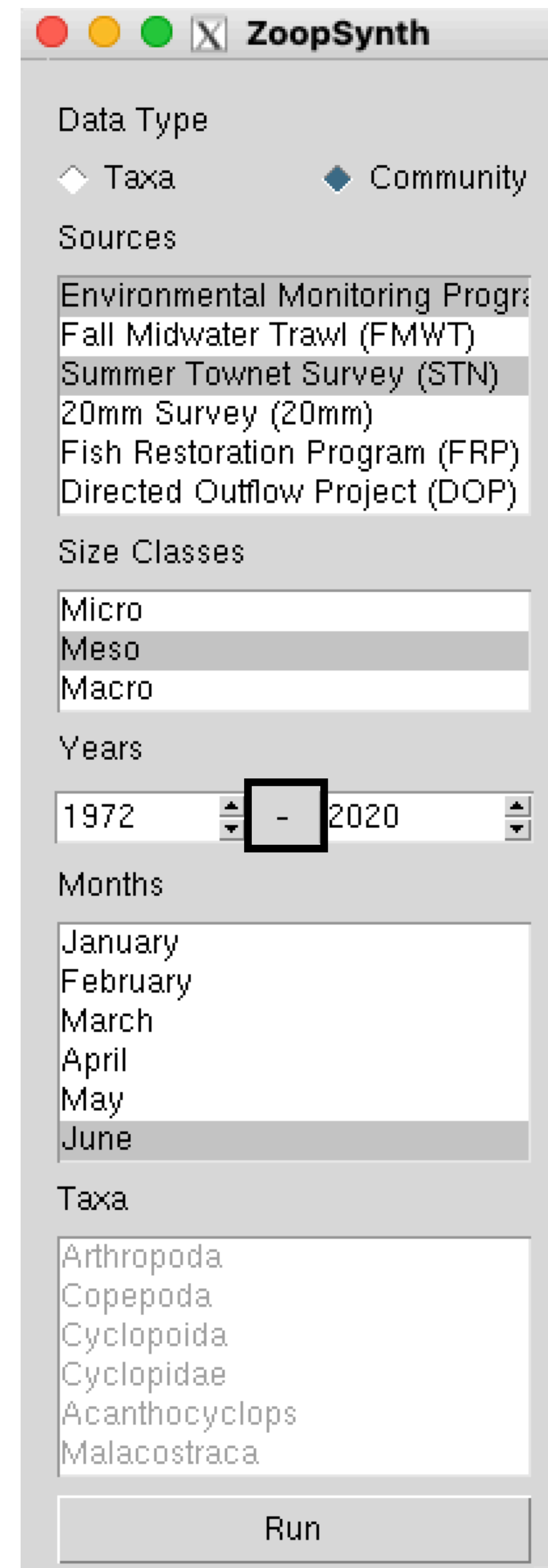
Commands

```
tkgrid(tkbutton(mainframe, text = "Run", command = fetch_and_plot),
       row = 12, column = 0, columnspan = 3, sticky = "we", padx = 5, pady = 5)
```

ZoopSynth

Data Type

◇ Taxa   ◆ Community

Sources

Environmental Monitoring Progra
Fall Midwater Trawl (FMWT)
Summer Townet Survey (STN)
20mm Survey (20mm)
Fish Restoration Program (FRP)
Directed Outflow Project (DOP)

Size Classes

Micro
Meso
Macro

Years

1972 ⇅ - 2020 ⇅

Months

January
February
March
April
May
June

Taxa

Arthropoda
Copepoda
Cyclopoida
Cyclopidae
Acanthocyclops
Malacostraca

Run

Commands

```r
tkgrid(tkbutton(mainframe, text = "Run", command = fetch_and_plot),
       row = 12, column = 0, columnspan = 3, sticky = "we", padx = 5, pady = 5)


fetch_and_plot <- function(...){
  fetch()
  plot_samples(zoop_data, source_colors)
}
```

A command function should take no arguments (except …) and should not return values.

Commands

```r
tkgrid(tkbutton(mainframe, text = "Run", command = fetch_and_plot),
       row = 12, column = 0, columnspan = 3, sticky = "we", padx = 5, pady = 5)

fetch <- function(...){
  tx = NULL
  if (tclvalue(datatype) == "Taxa"){
    tx = completeTaxaList[as.numeric(tkcurselection(taxa_lb)) + 1]
  }

  zoop_data <<- Zoopsynther(
    Data_type = tclvalue(datatype),
    Sources = source_codes[as.numeric(tkcurselection(sources_lb)) + 1],
    Size_class = size_codes[as.numeric(tkcurselection(sizes_lb)) + 1],
    Taxa = tx,
    Months = as.numeric(tkcurselection(months_lb)) + 1,
    Years = as.numeric(tclvalue(min_yr)):as.numeric(tclvalue(max_yr)),
    All_env = FALSE) |>
    prep_samples()
}
```

Commands

```r
tkgrid(tkbutton(mainframe, text = "Run", command = fetch_and_plot),
       row = 12, column = 0, columnspan = 3, sticky = "we", padx = 5, pady = 5)

fetch <- function(...){
  tx = NULL
  if (tclvalue(datatype) == "Taxa"){
    tx = completeTaxaList[as.numeric(tkcurselection(taxa_lb)) + 1]
  }

  zoop_data <<- Zoopsynther(
    Data_type = tclvalue(datatype),
    Sources = source_codes[as.numeric(tkcurselection(sources_lb)) + 1],
    Size_class = size_codes[as.numeric(tkcurselection(sizes_lb)) + 1],
    Taxa = tx,
    Months = as.numeric(tkcurselection(months_lb)) + 1,
    Years = as.numeric(tclvalue(min_yr)):as.numeric(tclvalue(max_yr)),
    All_env = FALSE) |>
    prep_samples()
}
```

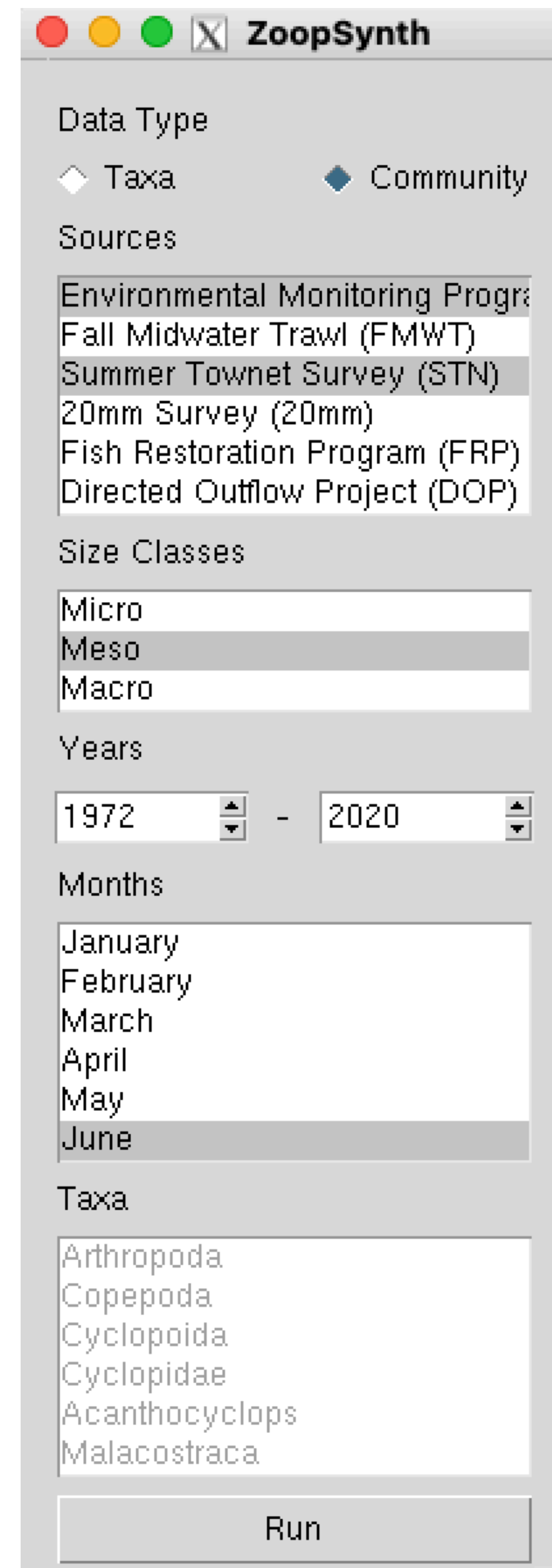Not best practice to use the super assignment operator (<<-).

Commands

```r
tkgrid(tkbutton(mainframe, text = "Run", command = fetch_and_plot),
       row = 12, column = 0, columnspan = 3, sticky = "we", padx = 5, pady = 5)

fetch <- function(...){
  tx = NULL
  if (tclvalue(datatype) == "Taxa"){
    tx = completeTaxaList[as.numeric(tkcurselection(taxa_lb)) + 1]
  }

  zoop_data <<- Zoopsynther(
    Data_type = tclvalue(datatype),
    Sources = source_codes[as.numeric(tkcurselection(sources_lb)) + 1],
    Size_class = size_codes[as.numeric(tkcurselection(sizes_lb)) + 1],
    Taxa = tx,
    Months = as.numeric(tkcurselection(months_lb)) + 1,
    Years = as.numeric(tclvalue(min_yr)):as.numeric(tclvalue(max_yr)),
    All_env = FALSE) |>
    prep_samples()
}
```
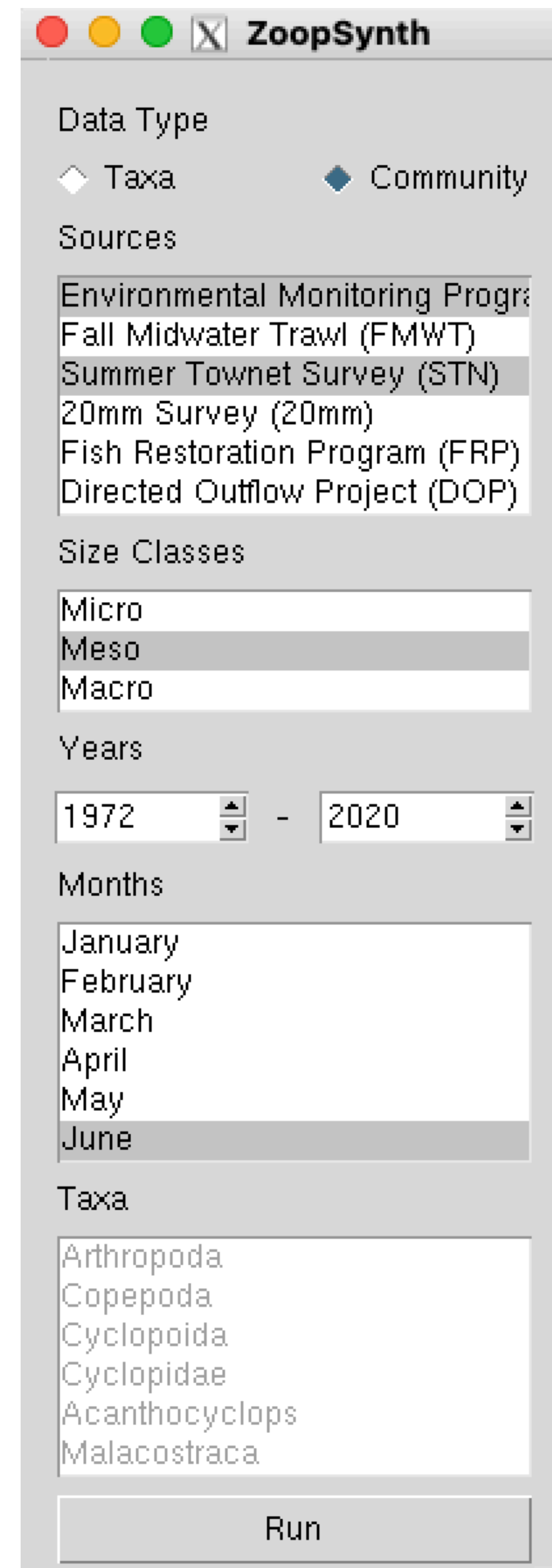
Retrieve current value from a tcl variable.

Commands

```r
tkgrid(tkbutton(mainframe, text = "Run", command = fetch_and_plot),
       row = 12, column = 0, columnspan = 3, sticky = "we", padx = 5, pady = 5)

fetch <- function(...){
  tx = NULL
  if (tclvalue(datatype) == "Taxa"){
    tx = completeTaxaList[as.numeric(tkcurselection(taxa_lb)) + 1]
  }

  zoop_data <<- Zoopsynther(
    Data_type = tclvalue(datatype),
    Sources = source_codes[as.numeric(tkcurselection(sources_lb)) + 1],
    Size_class = size_codes[as.numeric(tkcurselection(sizes_lb)) + 1],
    Taxa = tx,
    Months = as.numeric(tkcurselection(months_lb)) + 1,
    Years = as.numeric(tclvalue(min_yr)):as.numeric(tclvalue(max_yr)),
    All_env = FALSE) |>
    prep_samples()
}
```

Retrieve zero-based indices for selected months.
Zoopsynther() accepts numeric month; add one to
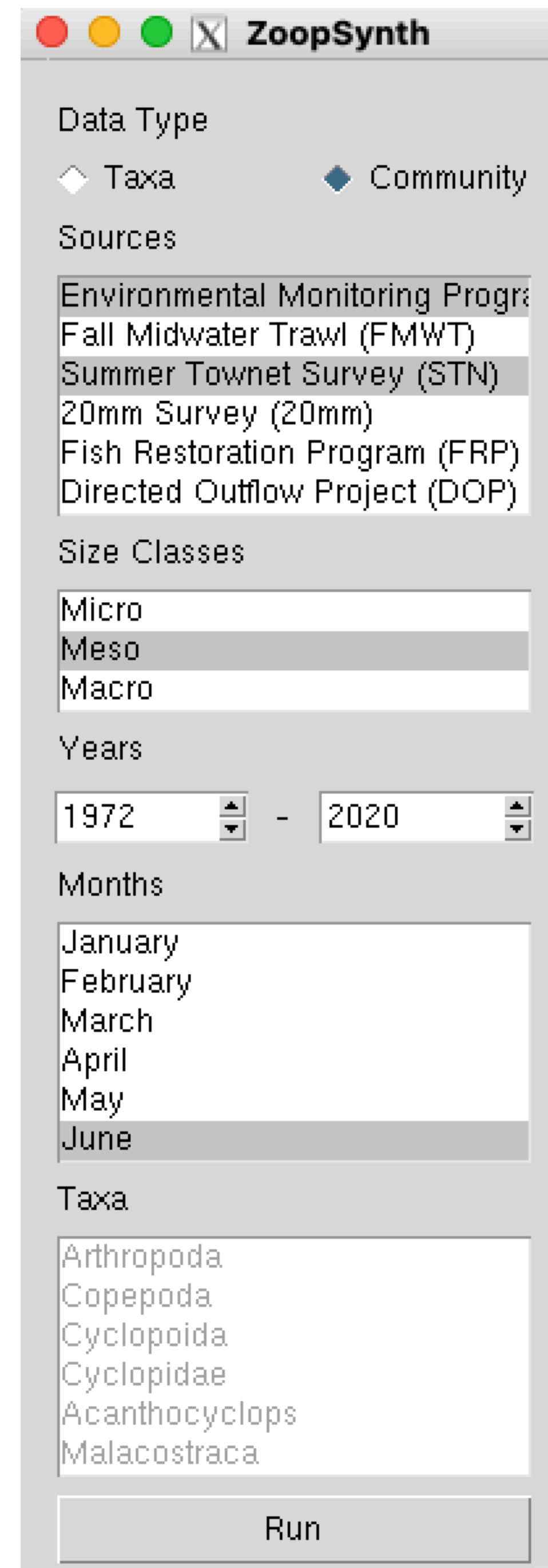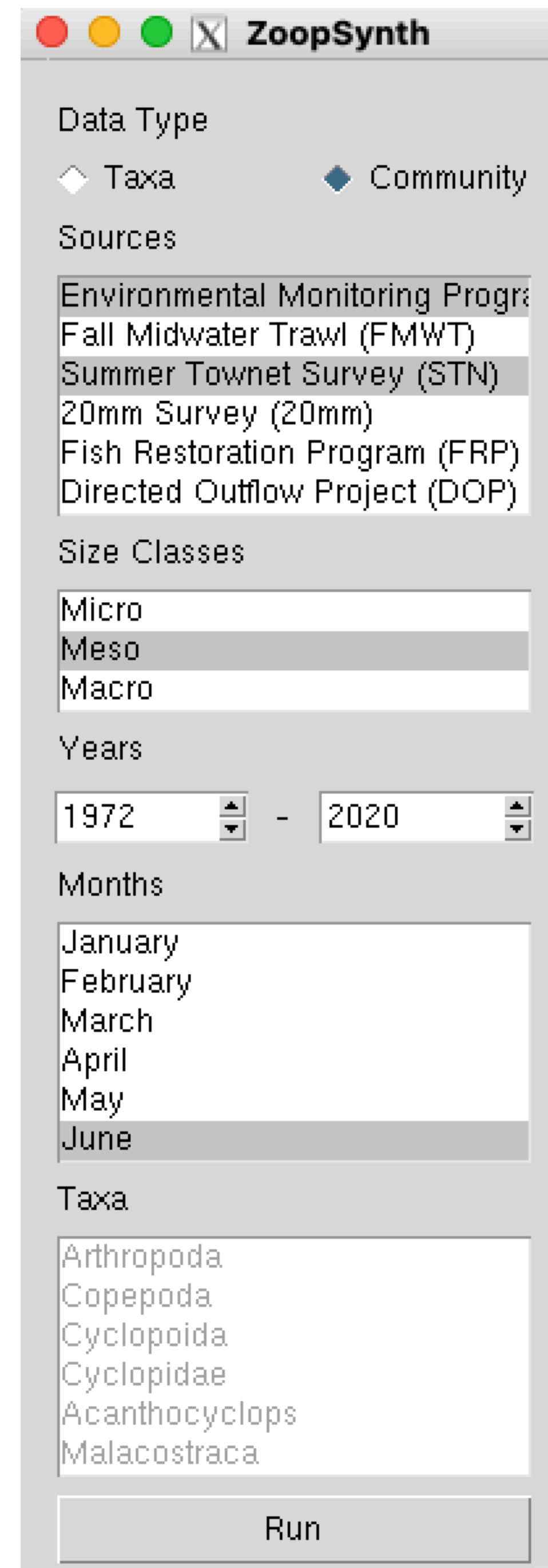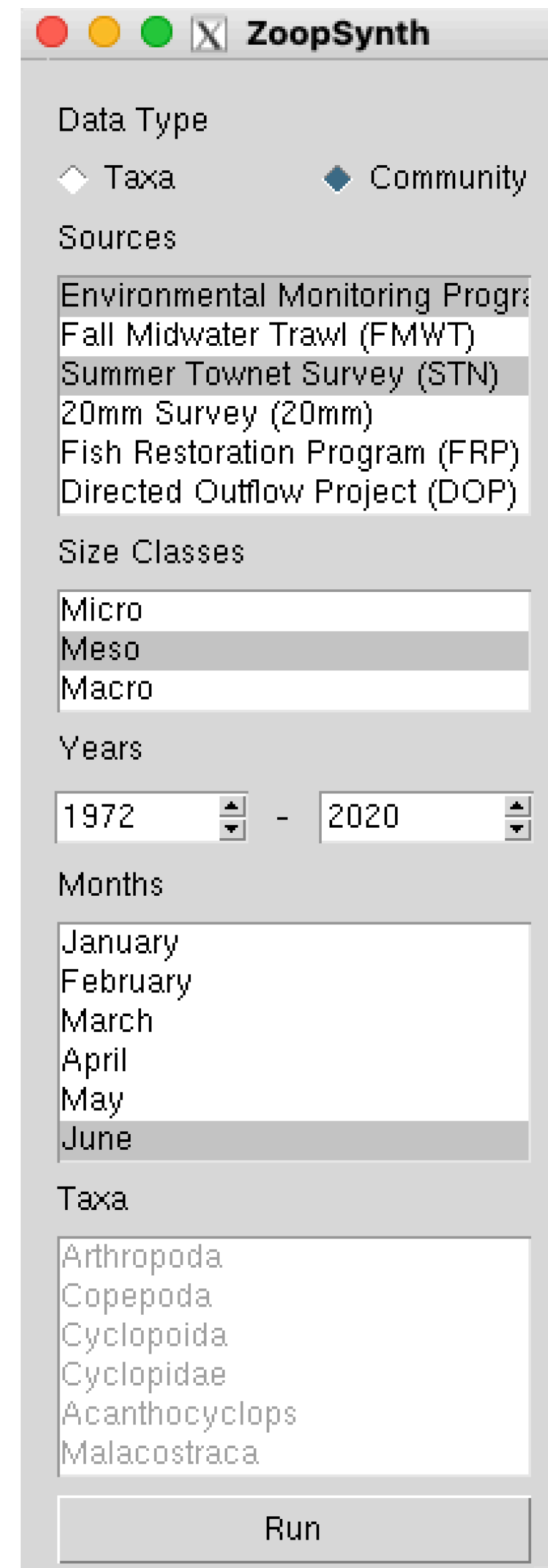indices to get months.

Commands

```
tkgrid(tkbutton(mainframe, text = "Run", command = fetch_and_plot),
       row = 12, column = 0, columnspan = 3, sticky = "we", padx = 5, pady = 5)

fetch <- function(...){
  tx = NULL
  if (tclvalue(datatype) == "Taxa"){
    tx = completeTaxaList[as.numeric(tkcurselection(taxa_lb)) + 1]
  }

  zoop_data <<- Zoopsynther(
    Data_type = tclvalue(datatype),
    Sources = source_codes[as.numeric(tkcurselection(sources_lb)) + 1],
    Size_class = size_codes[as.numeric(tkcurselection(sizes_lb)) + 1],
    Taxa = tx,
    Months = as.numeric(tkcurselection(months_lb)) + 1,
    Years = as.numeric(tclvalue(min_yr)):as.numeric(tclvalue(max_yr)),
    All_env = FALSE) |>
    prep_samples()
}
```

Retrieve zero-based indices for selected sources.
Zoopsynther() accepts source codes (e.g., EMP, STN);
use indices to get values from source_codes vector.

ZoopSynth

Data Type
◇ Taxa   ◆ Community

Sources

Environmental Monitoring Progra
Fall Midwater Trawl (FMWT)
Summer Townet Survey (STN)
20mm Survey (20mm)
Fish Restoration Program (FRP)
Directed Outflow Project (DOP)

Size Classes

Micro
Meso
Macro

Years

1972   -   2020

Months

January
February
March
April
May
June

Taxa

Arthropoda
Copepoda
Cyclopoida
Cyclopidae
Acanthocyclops
Malacostraca

Run

Plotting

```r
plot_samples <- function(data, source_colors){
  # need to open device window before plotting
  # set the device type based on the OS
  os = Sys.info()[['sysname']]
  if (os == "Darwin"){
    quartz(width = 12)
  } else if (os == "Windows") {
    windows(width = 12)
  } else {
    X11(width = 12)
  }

  plot(ggplot_samples(data, source_colors))
}
```

Plotting

```r
plot_samples <- function(data, source_colors){
    # need to open device window before plotting
    # set the device type based on the OS
    os = Sys.info()[['sysname']]
    if (os == "Darwin"){
      quartz(width = 12)
    } else if (os == "Windows") {
      windows(width = 12)
    } else {
      X11(width = 12)
    }

    plot(ggplot_samples(data, source_colors))
}
```
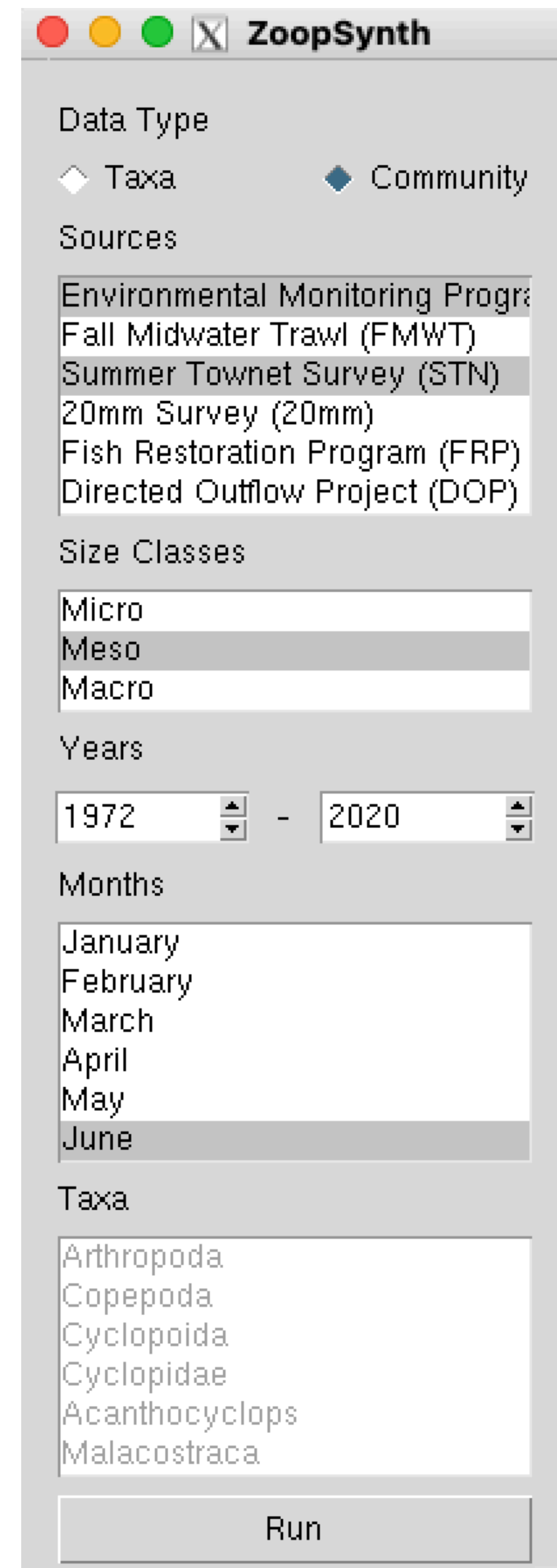
Need to call plot() on a ggplot object for it to be displayed in the device window.

tkrplot
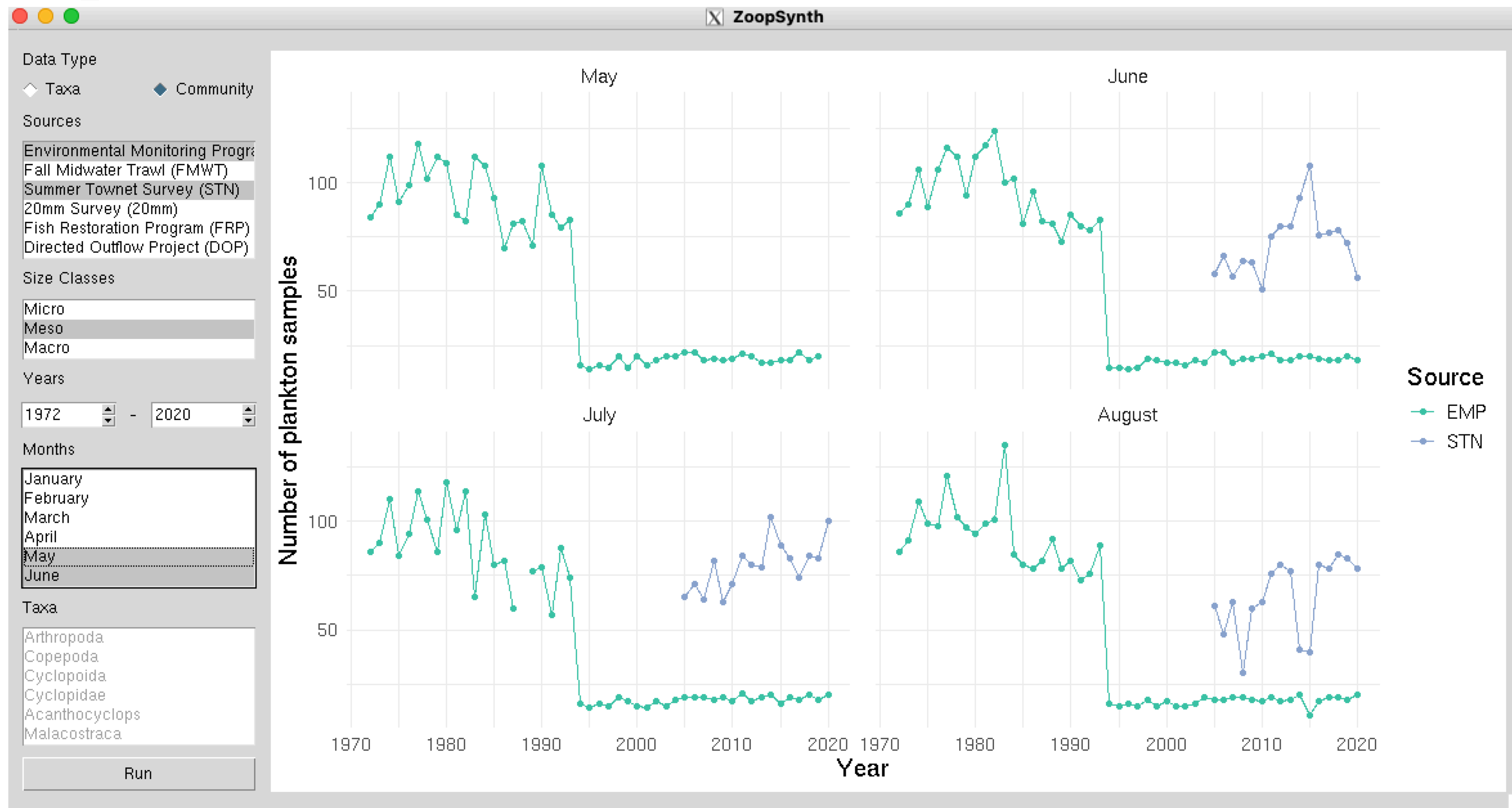
Documentation

Most of the functions in the **tcltk** package are really just a thin layer covering an underlying Tcl command. Converting all the Tcl/Tk documentation for R is a daunting task, so you have to make do with the help for Tcl/Tk itself. This is fairly easy once you get the hang of some simple translation rules.

Documentation

Most of the functions in the **tcltk** package are really just a thin layer covering an underlying Tcl command. Converting all the Tcl/Tk documentation for R is a daunting task, so you have to make do with the help for Tcl/Tk itself. This is fairly easy once you get the hang of some simple translation rules.

pathName **selection** *option arg*
> This command is used to adjust the selection within a listbox. It has several forms, depending on *option*:

pathName **selection anchor** *index*
> Sets the selection anchor to the element given by *index*. If *index* refers to a non-existent element, then the closest element is used. The selection anchor is the end of the selection that is fixed while dragging out a selection with the mouse. The index **anchor** may be used to refer to the anchor element.

pathName **selection clear** *first* ?*last*?
> If any of the elements between *first* and *last* (inclusive) are selected, they are deselected. The selection state is not changed for elements outside this range.

pathName **selection includes** *index*
> Returns 1 if the element indicated by *index* is currently selected, 0 if it is not.

pathName **selection set** *first* ?*last*?
> Selects all of the elements in the range between *first* and *last*, inclusive, without affecting the selection state of elements outside that range.

```
tkselection.set(sizes_lb, 1)
# if consecutive, can specify first and last index to select multiple
tkselection.set(months_lb, 5, 7)
```

Resources

- ZoopSynth-tcltk GitHub repo: https://github.com/hinkelman/ZoopSynth-tcltk

- Phil Spector's Class Notes (Ch. 12): https://www.stat.berkeley.edu/~spector/s133/all2011.pdf

- tcltk package includes four demos: tkcanvas, tkdensity, tkfaq, and tkttest

- tcltk primer: https://journal.r-project.org/articles/RN-2001-026/RN-2001-026.pdf

- Tk Commands documentation: https://www.tcl.tk/man/tcl8.6/TkCmd/contents.htm