```
#Load data
source('helper.R')

# View impacts table
View(impacts)


# prepare features
zn1 <- feat_stack * impacts[,"Protect"]
zn2 <- feat_stack * impacts[,"Restore"]
zn3 <- feat_stack * impacts[,"Manage"]
zn4 <- feat_stack * impacts[,"Urban_Green"]

# Create Zone file
zns <- zones("Protect" = zn1,
             "Restore" = zn2,
             "Manage" = zn3,
             "Urban_Green" = zn4,
             feature_names = names(zn1))

# View weights table
View(wgts)

weights.temp <- as.matrix(matrix(rep(wgts$weight, 4),
                                 ncol = 4,
                                 nrow = nlayers(feat_stack)))

pu_temp <- pu_all[["area"]][["avail"]]

p1 <- problem(pu_temp, zns) %>%
  add_max_utility_objective(c(Protect_target,
                              Restore_target,
                              Manage_target,
                              Urban_Green_target)) %>%
  add_feature_weights(weights.temp) %>%
  add_gurobi_solver(gap = 0)

if(locked_in){
  p1 <- p1 %>%
    add_locked_in_constraints(stack(Protect_locked,
                                    Restore_locked,
                                    Manage_locked,
                                    Urban_Green_locked))
}

#Solve the problem
s1 <- prioritizr::solve(p1, force = TRUE)
```

```
# View results
rst <- s1
names(rst) <- c("Protect",
                "Restore",
                "Manage",
                "Urban Green")

rst[rst == 0] <- NA
cols <- c('#4daf4a', '#984ea3', '#377eb8', '#e41a1c')


outl <- leaflet() %>% addTiles() %>%
  # Base groups
  addProviderTiles("Esri.WorldStreetMap",group = "StreetMap") %>%
  addProviderTiles("Esri.WorldImagery", group = "Aerial") %>%
  addProviderTiles("Stamen.Terrain", group = "Terrain")# %>%



# Overlay groups
for(ii in 1:nlayers(rst))
  outl <- addRasterImage(outl, rst[[ii]], colors = cols[ii], opacity =
0.9,
                         maxBytes = 8 * 1024 * 1024, group =
names(rst)[ii], project = FALSE)

outl <- addLayersControl(outl,
                         baseGroups = c("StreetMap", "Aerial",
"Terrain"),
                         overlayGroups = names(rst),
                         options = layersControlOptions(collapsed =
FALSE)
) %>%
  addLegend(colors = cols, labels = names(rst),
            title = "Legend") %>%
  addTiles(attribution = sprintf('<h5>Map created on %s via <a
href="http://forbasin.forestry.ubc.ca/CDFCP_prioritization/"
target="_blank">CDFCP conservation prioritization tool</a> developed
by <a href="mailto:mail@richard-schuster.com">Richard Schuster</a> for
the <a href="http://arcese.forestry.ubc.ca/marxan-tool-cdfcp/"
target="_blank">Aecese Lab</a>.</h5>',Sys.Date()))

outl
```