```
#Add libraries/packages that we will need
library(sp)
library(rgdal)
library(raster)


#Setting a seed to ensure that the simulated random values can be
replicated
set.seed(0.0001)
#Getting numbers within the longitude extents of Kenya
# Get a vector of 3 numbers from 0 to 100
lon <- runif(50, min=35, max=37)
lat <- runif(50, min= -2, max = 2)
var <- runif(50, min= 0, max = 100)

#Creating a data frame
data <- data.frame (lon,lat,var)

#getting summary statistics of the data
head (data) # displays a few rows of the data from the top/head of the
file
tail(data) # displays a few rows from the bottom/tail

summary(data) # provides a summary statistic of all the
variables/columns of the data

#You can also access the column names
colnames(data)

#now let's try and plot thge data
plot(lon, lat, data = data)

#add title and axes labels to the plot
plot(lon, lat, data = data, main = "Plotting Data", xlab =
"Longitude", ylab="Latitude")

#we can improve the plot by changing the sizes of points by variable
plot(lon, lat, data = data, cex = var/100, col = "red", main =
"Plotting Data", xlab = "Longitude", ylab="Latitude")

#Note that although we are plotting longitudes and latitudes, we have
not defined them as spatial objects

#We can create spatial objects by using sp package
#first create a coordinates dataframe
coords <- data.frame(lon,lat, data = data)

#convert the coordinates to SpatialPoints

dataSP <- SpatialPoints(coords = coords, proj4string =
CRS("+init=epsg:4326"))

#now you have created spatial point data
dataSP
```

```
#Notice that apart from cordinates, no attributes have been added to
the data
#We can add these using SpatialPointsDataFrame
variable <- data.frame(var) # Changing data column to a data frame
dataSPDF = SpatialPointsDataFrame(coords, data = variable, proj4string
= CRS("+init=epsg:4326"))

spplot(dataSPDF, col = "variable")

points <- readOGR("ken_random_pts.GeoJSON")
elevation <- raster::getData("alt", country="KEN")
elev <- raster::extract(elevation, points)

pointsData <- data.frame(points@data, elev)
write.csv(pointsData, "sample_points.csv")
counties <- readOGR("kenya_counties.GeoJSON")
plot(counties, col = 'grey', border = "white")

counties <- readOGR("kenya_counties.GeoJSON")

# Plotting both country and data points
plot(counties, col = 'grey', border = "white")
```