# Relatório de Mapas Geográfico

**Consultor Responsável**

Gabriel Lopes,João Pedro Malta, Rafaela Maciel, Tiago Sampaio

**Requerente**

Guilherme Rodrigues

Brasília, 1º de Novembro de 2019

# Contents

November 6, 2019

# 1 Introdução

**Para a apresentação do trabalho, utilizamos o pacote "Leaflet" do RStudio, o qual permite a criação de mapas interativos com base em dados da internet. Não agindo apenas sozinho, o R possui uma comunicação com o Google, permitindo que utilize seus mapas e os manipule, criando outros muito interessantes. Além de fornecer uma aba muito maior de criação e coloração de mapas. Após instalar e salvar o pacote, o código mais simples para criarmos um mapa é o abaixo:**

```
m <- leaflet() %>%
addTiles() %>% #Criar um mapa básico
addMarkers(lng=-47.869595, lat=-15.762657,
           popup="Departamento de Matemática")
```

**(Os mapas estarão nos slides disponíveis em nosso repositório do GitHub, em um README.file).**

# 2 Etiquetas

**As etiquetas e os pop-ups funcionam de forma muito parecida. As etiquetas são um conteúdo textual que se comportam como marcadores e aparecerão sempre em seu mapa, são fixos. Já os pop-ups também são conteúdos textuais em forma de marcadores, porém não ficam aparecendo no mapa. A única coisa que fica a mostra é a localização vinculada ao pop-up e, ao clicá-la, o texto configurado irá aparecer.**

```
library(htmltools)
df <- read.csv(textConnection(
  "Name, Lat, Long
"Departamento de Estatística",-15.758656,-47.869035
"ICC",-15.763306,-47.869685
"Restaurante Universitario",-15.764253,-47.870273"))
leaflet(df) %>% addTiles() %>%
  addMarkers(~Long, ~Lat, label = ~htmlEscape(Name))
```

**É possível personalizar os rótulos dos marcadores usando o labelOptions. Mudando o tamanho e o texto apenas utilizando um CSS customizado.**

```
leaflet() %>% addTiles() %>% setView(-47.871149,-15.761568, 16) %>%
  addMarkers(
    lng = -47.869035, lat = -15.758656,
    label = "Popup com Fonte 15px",
    labelOptions = labelOptions(noHide = T, textsize = "15px")) %>%
  addMarkers(
    lng = -47.871453, lat = -15.760986,
    label = "Popup com estilo CSS",
    labelOptions = labelOptions(noHide = T, direction = "bottom",
                                style = list(
                                "color" = "red",
                                "font-family" = "serif",
                                "font-style" = "italic",
```

```
                    "box-shadow" = "3px 3px rgba(0,0,0,0.25)",
                    "font-size" = "12px",
                    "border-color" = "rgba(0,0,0,0.5)"
                )))
```

# 3 Círculos

**Os cículos, ao invés de posicionarmos marcadores em somente um ponto exato, uma área de um círculo é posta ao redor das latitudes e longitudes especificadas, especificando o tamanho do círculo logo após as coordenadas.**

```
cities <- read.csv(textConnection("City,Lat,Long,Pop
Departamento de Estatística,-15.758656,-47.869035,3
ICC,-15.763306,-47.869685,10
Restaurante Universitario,-15.764253,-47.870273,5"))
leaflet(cities) %>% addTiles() %>%
  addCircles(lng = ~Long, lat = ~Lat, weight = 1,
             radius = ~sqrt(Pop) * 30, popup = ~City
  )
```

# 4 Retângulos

**Os retângulos seguem a mesma lógica dos círculos, porém, informaremos ao R duas latitudes e duas longitudes, criando o retângulo com essas 4 linhas imaginárias fornecidas, usando a função addRectangles().**

```
leaflet() %>% addTiles() %>%
addRectangles(
  lat1=-15.765073, lng1=-47.866681,
  lat2=-15.760592, lng2= -47.871563,
  fillColor = "transparent"
)
```

# 5 Cores em Mapas

**Cores podem ser adicionadas como classificação das variáveis, possibilitando melhores e importantes informações sobre os mapas. São apresentadas quatro funções para colorir mapas, sendo três para dados contínuos e uma para dados categóricos. Todas essas possuem dois parâmetros em comum. O parâmetro "palette" especifica as cores para as quais os dados são mapeados, já o parâmetro "domain" informa à função de cor o intervalo dos valores de entrada. Há a possibilidade de passar NULL para criar uma função de paleta que não tenha um intervalo predefinido, o intervalo será inferido a partir dos dados sempre que você chamar a função da paleta.**

- Função colorNumeric: Para dados contínuos e saída das cores no mapa em forma contínua (com vários tons, dando essa percepção);
- Função colorBin: Para dados contínuos e saída das cores no mapa em forma discreta (com somente um tom de cor, podendo especificar a entrada das cores ou não);
- Função colorQuantile: Para dados contínuos e saída das cores no mapa em forma discreta (com somente um tom de cor, podendo especificar a entrada das cores ou não);
- Função colorFactor: Para dados categóricos e cada cor se relacionando com uma categoria dentro dos dados, sendo possível informar a quantidade de cores por categoria ou deixando a função interpolar as cores e as categorias sozinha;

```
  # Chamamos a função colorNumeric para criar uma função no R
# para a paleta de cores e definimos seu domínio;
 pal <- colorNumeric(c("red", "green", "blue"), 1:10)
# Convertemos a paleta para um vetor e obtemos as cores
# pal(c(1,6,9))
```

# 6 Tipos de Mapas

**O Leaflet suporta mapas base usando blocos de mapas , popularizados pelo Google Maps e agora usados por quase todos os mapas interativos da web. Como alternativa, muitos mapas base populares gratuitos de terceiros podem ser adicionados usando a função ”addProviderTiles()”, que é implementada usando o plugin de fornecedores de folhetos. Por conveniência, o folheto também fornece uma lista nomeada de todos os fornecedores de blocos de terceiros que são suportados pelo plug-in. Tais temas se encontram disponíveis no link: http://leaflet-extras.github.io/leaflet-providers/preview/index.html**

**Também existe a possibilidade de sobrepormos dois ou mais mapas, adicionando camadas em formas de mosaico e definindo sua opacidade, ademais de existirem os mapas interativos, os quais podemos alterar seus parâmetros no próprio mapa e ele mudará sua cor, a área demarcada ou a posição de círculos, por exemplo(para ver o mapa interativo da apresentação, ver o slide disponível no github).**

# 7 Chave API e Google Way

**A chave API é uma chave de identificação (código) que permite autenticar métodos que estão na documentação API, possibilitando, dentro do pacote GoogleWay, a utilização da pesquisa do Google pelas localizações. Ao invés de selecionarmos as coordenadas e colocarmos em mapas, como antes era feito, insere-se no código o nome a ser pesquisado(por exemplo, academias de Brasília) e com a vinculação do R com o Google, será retornado, em um mapa, todas as academias registradas de Brasília.**