API

June 7, 2024

[1]:

```
install.packages("httr")
install.packages("rvest")
install.packages("stringr")
install.packages("vctrs")
```

```
Updating HTML index of packages in '.Library'
Making 'packages.html' … done
Updating HTML index of packages in '.Library'
Making 'packages.html' … done
Updating HTML index of packages in '.Library'
Making 'packages.html' … done
Updating HTML index of packages in '.Library'
Making 'packages.html' … done
```

[2]:

```
# Check if need to install rvest` library
require("httr")

library(httr)
```

```
Loading required package: httr
```

[3]:

```
# URL for Current Weather API
current_weather_url <- 'https://api.openweathermap.org/data/2.5/weather'
```

[4]:

```
# need to be replaced by your real API key
your_api_key <- "af944eac365499ade4d4dda48108bcee"
# Input `q` is the city name
# Input `appid` is your API KEY,
# Input `units` are preferred units such as Metric or Imperial
current_query <- list(q = "Seoul", appid = your_api_key, units="metric")
```

[5]:

```
response <- GET(current_weather_url, query = current_query)
```

[6]:

```
http_type(response)
```

'application/json'

[7]:

```
json_result <- content(response, as = "parsed")
```

[8]:

```
class(json_result)
```

'list'

[9]:

```
json_result
```

```
$coord  $lon  126.9778
        $lat  37.5683
$weather    1. $id  800
               $main  'Clear'
               $description  'clear sky'
               $icon  '01n'
$base  'stations'
$main  $temp  5.96
        $feels__like  5.96
        $temp__min  5.76
        $temp__max  7.69
        $pressure  1018
        $humidity  100
$visibility  10000
$wind  $speed  1.03
        $deg  20
$clouds  $all = 0
$dt  1697489903
$sys  $type  1
        $id  8105
        $country  'KR'
        $sunrise  1697492470
        $sunset  1697532821
$timezone  32400
$id  1835848
$name  'Seoul'
$cod  200
```

[ ]:

```
weather <- c()
visibility <- c()
temp <- c()
temp_min <- c()
temp_max <- c()
pressure <- c()
humidity <- c()
wind_speed <- c()
wind_deg <- c()
```

[10]:

```
# Create some empty vectors to hold data temporarily
# City name column
city <- c()
# Weather column, rainy or cloudy, etc
weather <- c()
# Sky visibility column
visibility <- c()
# Current temperature column
temp <- c()
# Max temperature column
temp_min <- c()
# Min temperature column
temp_max <- c()
# Pressure column
pressure <- c()
# Humidity column
humidity <- c()
# Wind speed column
wind_speed <- c()
# Wind direction column
wind_deg <- c()
# Forecast timestamp
forecast_datetime <- c()
# Season column
# Note that for season, you can hard code a season value from levels Spring,␣
 ↪Summer, Autumn, and Winter based on your current month.
season <- c()
```

[11]:

```
# Get forecast data for a given city list
get_weather_forecaset_by_cities <- function(city_names){
df <- data.frame()
    for (city_name in city_names){
        # Forecast API URL
        forecast_url <- 'https://api.openweathermap.org/data/2.5/forecast'
        # Create query parameters
        forecast_query <- list(q = city_name, appid =␣
 ↪"af944eac365499ade4d4dda48108bcee", units="metric")
```

```r
        # Make HTTP GET call for the given city
        response <- GET(forecast_url, query = forecast_query)
        # Note that the 5-day forecast JSON result is a list of lists. You can print the reponse to check the results
        #results <- json_list$list
        json_list <- content(response, as="parsed")
        results <- json_list$list
        # Loop the json result
        for(result in results) {
            city <- c(city, city_name)
            weather <- c(weather, result$weather[[1]]$main)
            # Get Visibility
            visibility <- c(visibility, result$visibility)
            # Get current temperature
            temp <- c(temp, result$main$temp)
            # Get min temperature
            temp_min <- c(temp_min, result$main$temp_min)
            # Get max temperature
            temp_max <- c(temp_max, result$main$temp_max)
            # Get pressure
            pressure <- c(pressure, result$main$pressure)
            # Get humidity
            humidity <- c(humidity, result$main$humidity)
            # Get wind speed
            wind_speed <- c(wind_speed, result$wind$speed)
            # Get wind direction
            wind_deg <- c(wind_deg, result$wind$deg)
            # Forecast timestamp
            forecast_datetime <-c(forecast_datetime, result$dt_txt)

            forecast_timestamp <- result$dt_txt
            hour <- as.numeric(strftime(forecast_timestamp, format="%H"))
            month <- as.numeric(strftime(forecast_timestamp, format="%m"))

            season <- "Spring"
            # Simple rule to determine season
            if (month >= 3 && month <= 5)
                season <- "Spring"
            else if(month >= 6  &&  month <= 8)
                season <- "Summer"
                else if (month >= 9  && month <= 11)
                    season <- "Autumn"
                else
                    season <- "Winter"

        }
        # Add the R Lists into a data frame
df <- data.frame(city = city,
                         weather = weather,
                         visibility = visibility,
                         temp = temp,
                         temp_min = temp_min,
                         temp_max = temp_max,
                         pressure = pressure,
                         humidity = humidity,
                         wind_speed = wind_speed,
                         wind_deg = wind_deg,
                         forecast_datetime = forecast_datetime,
                         season = season)
    }

    # Return a data frame
     return(df)

}
```

[12]:

```
# Check the generated data frame
print(df)
```

```
function (x, df1, df2, ncp, log = FALSE)
{
    if (missing(ncp))
        .Call(C_df, x, df1, df2, log)
    else .Call(C_dnf, x, df1, df2, ncp, log)
}
<bytecode: 0x55fa4131c928>
<environment: namespace:stats>
```

[13]:

```
cities <- c("Seoul", "Washington, D.C.", "Paris", "Suzhou")
cities_weather_df <- get_weather_forecaset_by_cities(cities)
head(cities_weather_df)
```

A data.frame: 6 × 12

	city	weather	visibility	temp	temp_min	temp_max	pressure	humidity
	<fct>	<fct>	<int>	<dbl>	<dbl>	<dbl>	<int>	<int>
1	Seoul	Clear	10000	8.38	8.38	13.21	1019	79
2	Seoul	Clear	10000	13.62	13.62	17.45	1019	49
3	Seoul	Clear	10000	19.14	19.14	19.14	1018	24
4	Seoul	Clear	10000	16.87	16.87	16.87	1019	34
5	Seoul	Clear	10000	15.15	15.15	15.15	1020	42
6	Seoul	Clouds	10000	14.08	14.08	14.08	1021	45

[14]:

```
# Write cities_weather_df to `cities_weather_forecast.csv`
write.csv(cities_weather_df, "cities_weather_forecast.csv", row.names=FALSE)
```

[15]:

```
##TASK: Download datasets as csv files from cloud storage
# Download several datasets

# Download some general city information such as name and locations
url <- "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
 ↪IBMDeveloperSkillsNetwork-RP0321EN-SkillsNetwork/labs/datasets/
 ↪raw_worldcities.csv"
# download the file
download.file(url, destfile = "raw_worldcities.csv")

# Download a specific hourly Seoul bike sharing demand dataset
url <- "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
 ↪IBMDeveloperSkillsNetwork-RP0321EN-SkillsNetwork/labs/datasets/
 ↪raw_seoul_bike_sharing.csv"
# download the file
download.file(url, destfile = "raw_seoul_bike_sharing.csv")
```

[ ]:

```

```