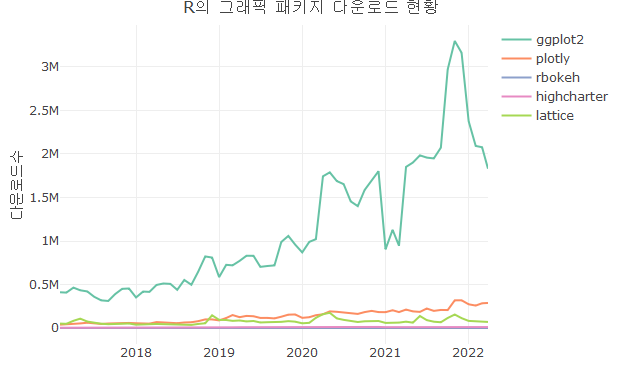
1. 데이터 시각화를 위한 plotly

R을 다루는 많은 교육코스나 서적에서 데이터의 시각화는 대부분 R base에서 제공하는 함수를 사용하거나 ggplot2 패키지를 사용하여는 방법을 위주로 설명한다. 이 두가지 방법은 데이터 시각화 결과가 우수한 편이기 때문에 많이 사용되고 있지만 정적(Static) 시각화이다. 정적 시각화는 최근 인포그래픽(Infographic)이라고 불리며 일반적으로 문서나 인쇄물에 많이 사용되고 웹에 게시되는 이미지로 사용된다. 그렇기 때문에 대부분 png, jpg, pdf 등의 벡터 혹은 픽셀 이미지 파일로 제공된다. 정적 데이터 시각화는 데이터 분석가의 의도에 맞춰 작성되기 때문에 데이터 분석가의 분석에 의존적일 수 밖에 없으며 독자의 의도에 따른 해석은 매우 제한될 수 밖에 없다.

이러한 제한점을 극복하기 위해 사용되는 데이터 시각화 방법이 동적(Dynamic) 시각화 혹은 인터랙티브(Interactive) 시각화라고 하는 방법이다. 이 동적 시각화는 시각화를 사용하는 사용자의 의도에 따라 데이터를 다각적 관점에서 살펴볼 수 있다는 점이 동적 시각화와 가장 크게 차이나는 점이다. 사용자의 의도에 따라 데이터가 동적으로 변동되어야 하기 때문에 데이터 시각화에 사용되는 매체는 인쇄물 형태 매체가 불가능하고 웹을 통해 제공한다. 따라서 일반적으로 동적 시각화는 웹 사이트에서 제공하는 대시보드(DashBoard)의 형태로 제공되는 것이 일반적이기 때문에 동적 시각화를 위해서는 동적 시각화 전용 패키지를 사용해 시각화 객체를 만드는 방법이외에 shiny 등의 패키지를 사용하여 대시보드를 만드는 것도 같이 익혀야 한다는 어려움이 따른다. R에서 동적 시각화를 위해 제공되는 패키지는 plotly, rbokeh, highcharter 등이 제공된다. 다음의 R 그래픽 패키지 다운로드 현황에서 보듯이 여전히 R에서 가장 많이 사용되고 있는 그래픽 패키지는 ggplot2이지만 최근 들어 그 사용량이 줄고 있고 R에서 많이 사용되던 lattice패키지가 점차 줄고 있다. 반면 plotly는 2021년 하반기부터 다운로드가 늘고 있고 다른 동적 그래픽 패키지에 비해서는 압도적인 다운로드 수를 보인다.

library(dlstats)  
  
#x <- cran\_stats(c("ggplot2", "plotly", "rbokeh", "highcharter"))  
x1 <- cran\_stats(c("ggplot2", "plotly", "rbokeh", "highcharter", 'lattice'))  
  
x1 |> plot\_ly() |>  
 add\_lines(x = ~end, y = ~downloads, color = ~package) |>  
 layout(title = 'R의 그래픽 패키지 다운로드 현황', xaxis = list(title = list(visible = F)), yaxis = list(title = '다운로드수'))

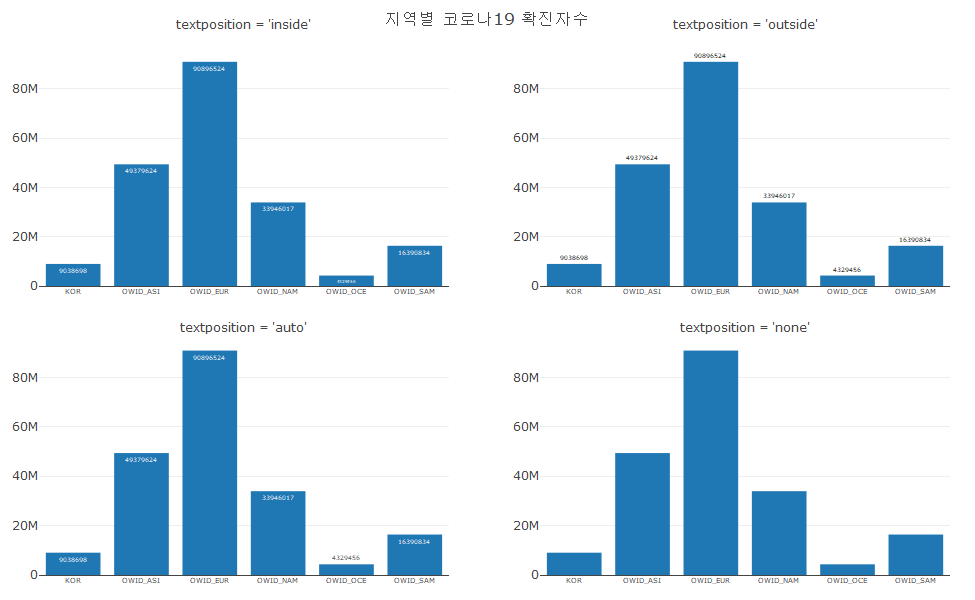


따라서 정적 시각화와 동적 시각화의 어느것이 더 효용성이 있는지를 단언할 수 없다. 데이터 시각화가 사용되는 매체, 데이터 시각화를 보는 대상, 데이터 시각화에서 보여주고자 하는 스토리에 따라서 정적 시각화를 사용해야 할 때와 동적 시각화를 사용해야 할 때를 적절히 선택해야 한다.

이번 장에서는 R에서 동적 시각화로 많이 사용되는 plotly 패키지를 사용하여 데이터를 시각화하는 방법을 알아본다. plotly를 사용한 데이터 시각화를 실습하기 위한 데이터는 ggplot2 실습에서 사용하던 취업통계 데이터(df\_취업통계)와 코로나바이러스 데이터 셋을 사용하도록 하겠다.

코로나바이러스 데이터 셋은 Github에서 매일 업데이트된 csv 파일을 다운로드 받았다. [[1]](#footnote-20) 이를 plotly 실습에 사용하기 위해 다음과 같이 전처리 하였다.

if(!require(readr)) {  
 install.packages('readr')  
 library(readr)  
}  
  
if(!require(lubridate)) {  
 install.packages('lubridate')  
 library(lubridate)  
}  
  
covid19\_df <- read\_csv(file = "D:/R/data/owid-covid-data.csv",  
 col\_types = cols(Date = col\_date(format = "%Y-%m-%d")  
 )  
 )  
  
covid19\_df <- read\_csv(file = "https://covid.ourworldindata.org/data/owid-covid-data.csv",  
 col\_types = cols(Date = col\_date(format = "%Y-%m-%d")  
 )  
 )  
  
  
covid19\_df\_100 <- covid19\_df |>   
 filter((iso\_code %in% c('KOR', 'OWID\_ASI', 'OWID\_EUR', 'OWID\_OCE', 'OWID\_NAM', 'OWID\_SAM'))) |>  
 filter(date >= today() - 100) |>   
 arrange(date)  
  
covid19\_df\_100\_wide <- covid19\_df\_100 |> select(date, location, new\_cases) |>   
 pivot\_wider(id\_cols = date, names\_from = location, values\_from = new\_cases) |>   
 arrange(date)  
  
## 공백을 쓰려면 ``  
names(covid19\_df\_100\_wide) <- c('date', 'Asia', 'Europe', 'North\_America', 'Oceania', 'South\_America', 'South\_Korea')  
  
covid19\_stat <- covid19\_df |> group\_by(iso\_code, continent, location) |>  
 summarise(인구수 = max(population, na.rm = T), 인당GDP = max(gdp\_per\_capita, na.rm = T),  
 전체확진자수 = sum(new\_cases, na.rm = T),  
 전체사망자수 = sum(new\_deaths, na.rm = T),   
 십만명당사망자수 = round(total\_deaths / population \*100000, 5),  
 십만명당중환자실 = last(icu\_patients\_per\_million),  
 재생산지수 = last(reproduction\_rate),  
 전체검사자수 = max(total\_tests, na.rm = T), new\_tests = sum(new\_tests, na.rm = T),  
 전체백신접종자수 = max(total\_vaccinations, na.rm = T),  
 백신접종자완료자수 = max(people\_fully\_vaccinated, na.rm = T),  
 부스터접종자수 = max(total\_boosters, na.rm = T),  
 백신접종완료률 = people\_fully\_vaccinated / population,  
 인구백명당백신접종완료률 = max(people\_fully\_vaccinated\_per\_hundred, na.rm = T),  
 인구백명당부스터접종자수 = max(total\_boosters\_per\_hundred, na.rm = T)  
 ) |> ungroup()  
  
margins <- list(t = 50, b = 25, l = 25, r = 25)



1. <https://github.com/owid/covid-19-data/blob/master/public/data/owid-covid-data.csv> [↑](#footnote-ref-20)