



공공 데이터 사용하기



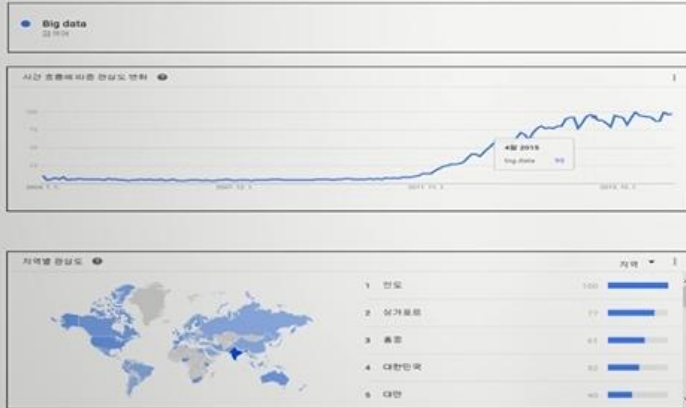
4차 산업혁명의 원동력 - 빅데이터

- 세상을 변화시키는 빅데이터
- Gartner
 - “데이터는 미래 경쟁력을 좌우하는 21세기 원유이다. 기업들은 다가오는 데이터 경제시대를 이해하고 이에 대비해야 한다”
- 빅데이터(BigData)
 - 오랜 시간 축적된 대용량의 데이터
 - 소프트웨어 기술
 - 부가가치를 찾아내는 기술

활용 사례로 본 빅데이터의 가치

❖ 구글 트렌드 (<http://www.google.co.kr/trends/>)

- 사용자들의 검색어와 시청 동영상 기반의 빅데이터 분석 서비스



❖ 에너지 및 자동차 분야

- 전기를 주동력 원으로 하는 자동차



포드(Ford)사 전기자동차의 정보를 관리하는 마이포드 모바일 앱

❖ 구글 북스 (<http://books.google.com/>)

- 인문학 분야에서 활용하기에 매우 좋은 빅데이터를 제공



❖ 공공 부문의 활용 사례

- 서울시에서 운영하고 있는 심야버스 노선
- KT의 통화량 통계 데이터와 서울시가 보유한 교통 데이터를 융합/분석



공공데이터 사이트

- 공공데이터 포털
 - 공공기관이 생성하거나 관리하고 있는 데이터
 - <https://www.data.go.kr/>
- 서울 열린 데이터 광장
 - <http://data.seoul.go.kr/>
 - 인기 정보 : 유동인구정보, 지하철역 승하차 인원, 공공 WIFI 위치 정보, 시장마트, 한강공원 공간정보
- 교통정보공개서비스
 - <http://openapi.its.go.kr/>
- 관광 데이터 TourApl
 - <http://api.visitkorea.or.kr/main.do>
- 부산시 빅데이터 포털
 - <http://bigdata.busan.go.kr>
- KOSIS 국가통계포털
 - <http://www.kosis.kr>
- 동남지방 통계청
 - <http://kostat.go.kr/office/dnro/index.action>
- 부산 공공데이터 포털
 - <http://data.busan.go.kr>

공공데이터를 활용한 자료 탐색

- 한강공원의 공간정보,
어디에 가면 자료를 찾을 수 있을까?
- 공공 와이파이가 되는 곳을 알고 싶는데,
검색으로 가능할까?
- 음식점 창업을 하고 싶는데,
우리 동네 모범음식점 현황을 한 눈에 알 수 있을까?

Example – 데이터 탐색





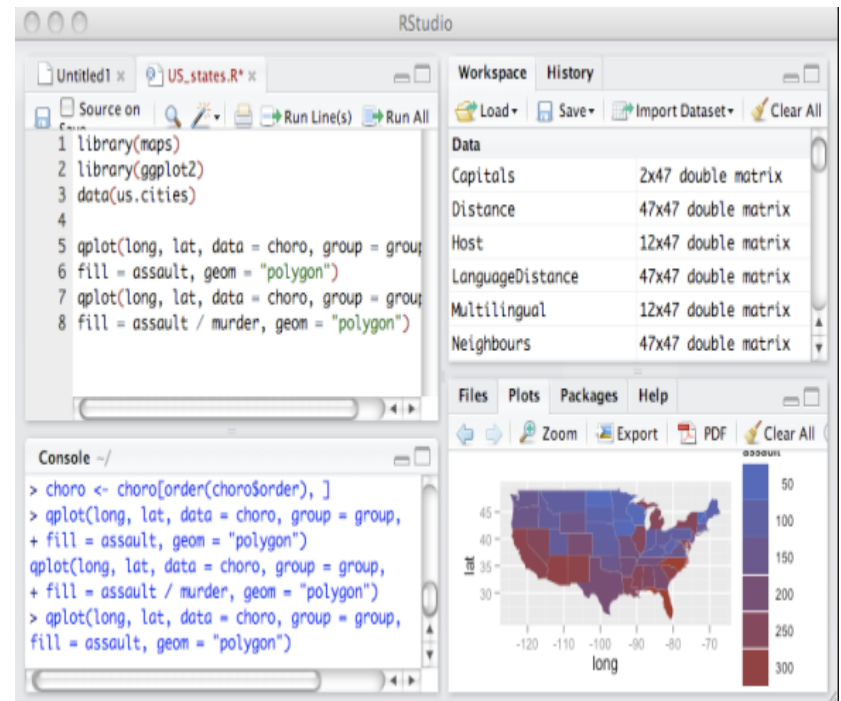
데이터 스토리텔링 with R

양자영

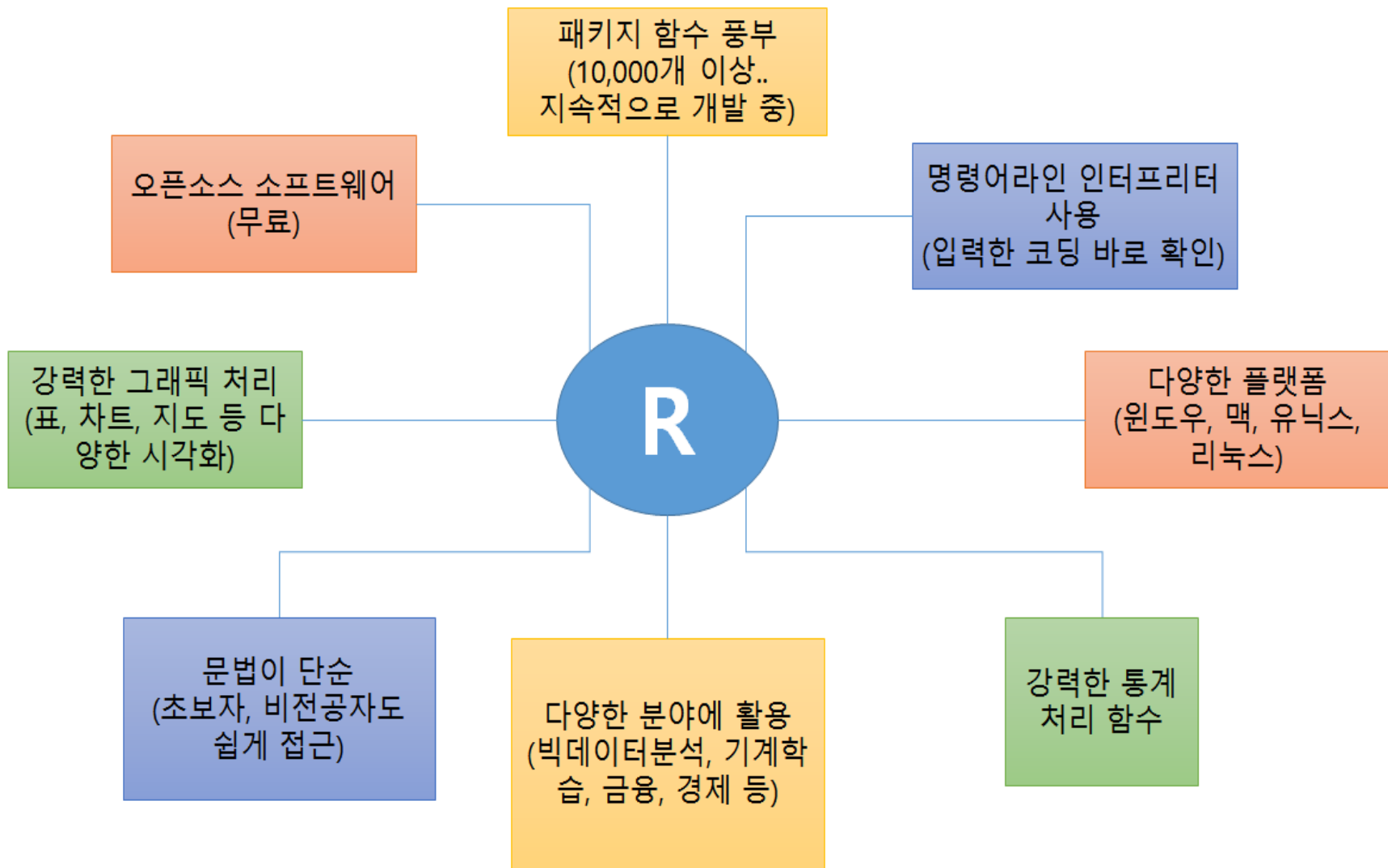


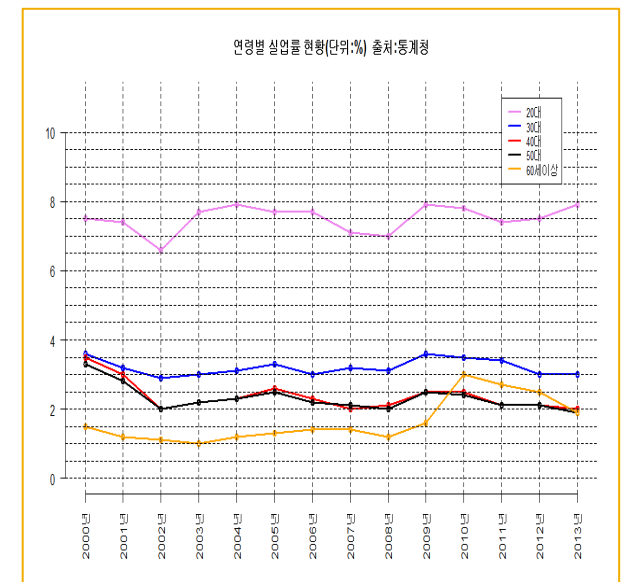
R언어

- 데이터 요약 및 시각화에 유리
- 패키지 및 함수 중심 프로그램 개발환경으로 몇몇의 함수의 연결로 실생활을 문제를 간략하게 해결
- 구글 등의 인터넷 및 공공자료를 쉽게 할 수 있어 자신의 분야의 정보를 빠르게 정리하여 활용하는데 도움이 됨



R 언어





프로그래밍을 위한 R환경

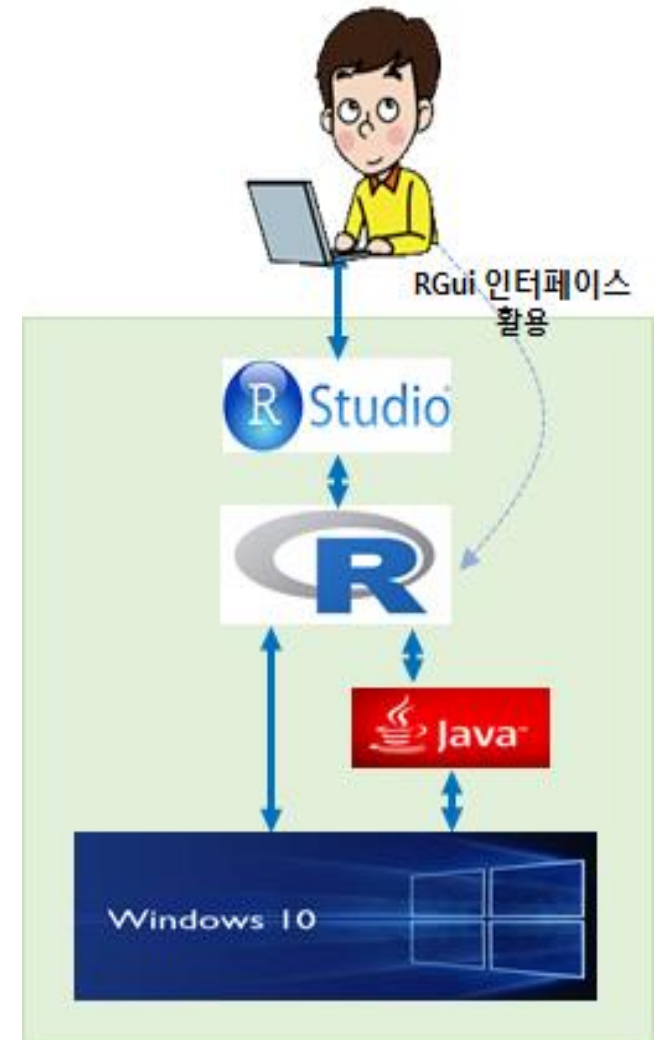
1. CRAN 사이트

(<https://cran.r-project.org/>)에서 다운로드 받아 설치

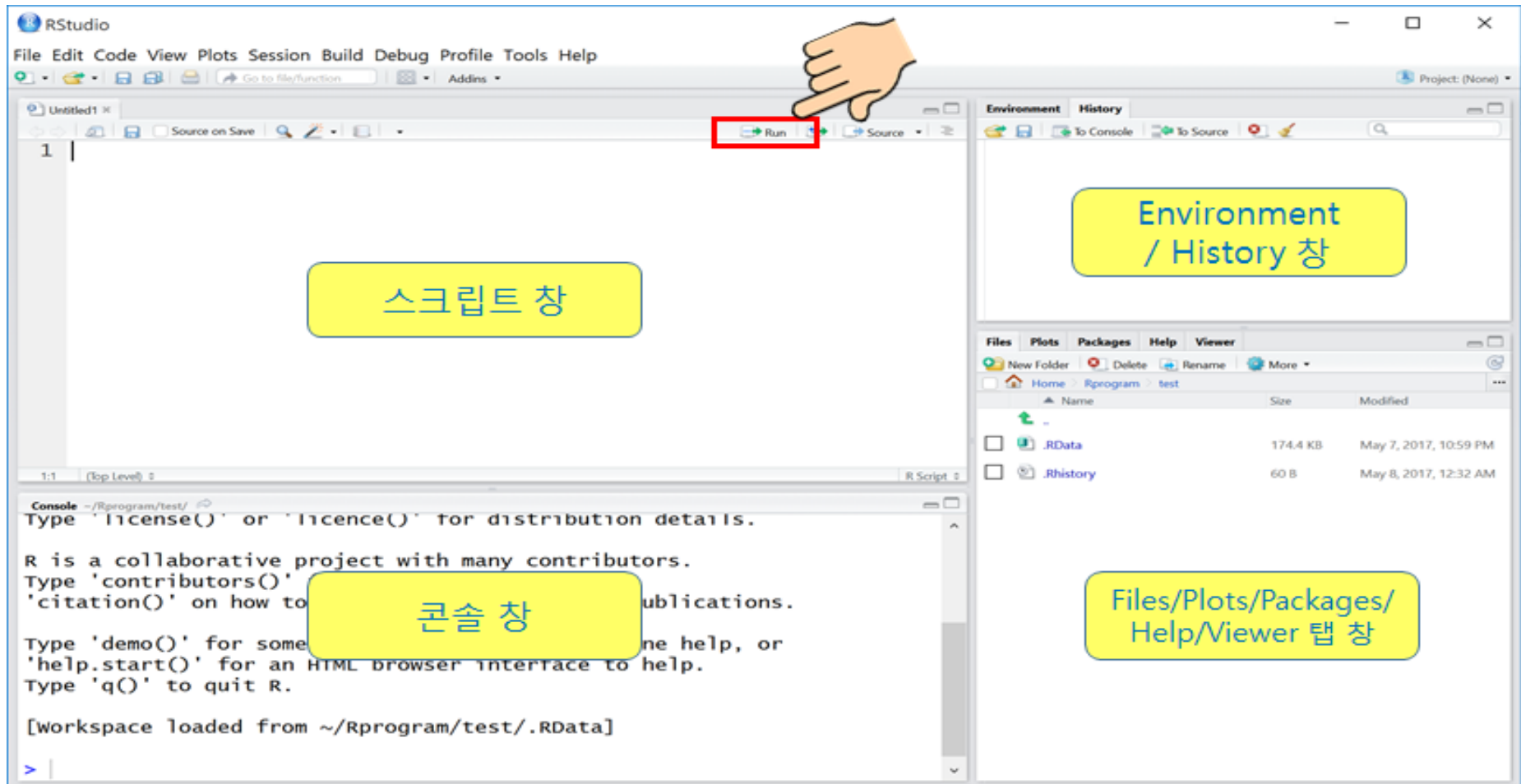
2. 더 편리한 사용자 인터페이스를 위해 RStudio를 설치

3. JAVA 설치

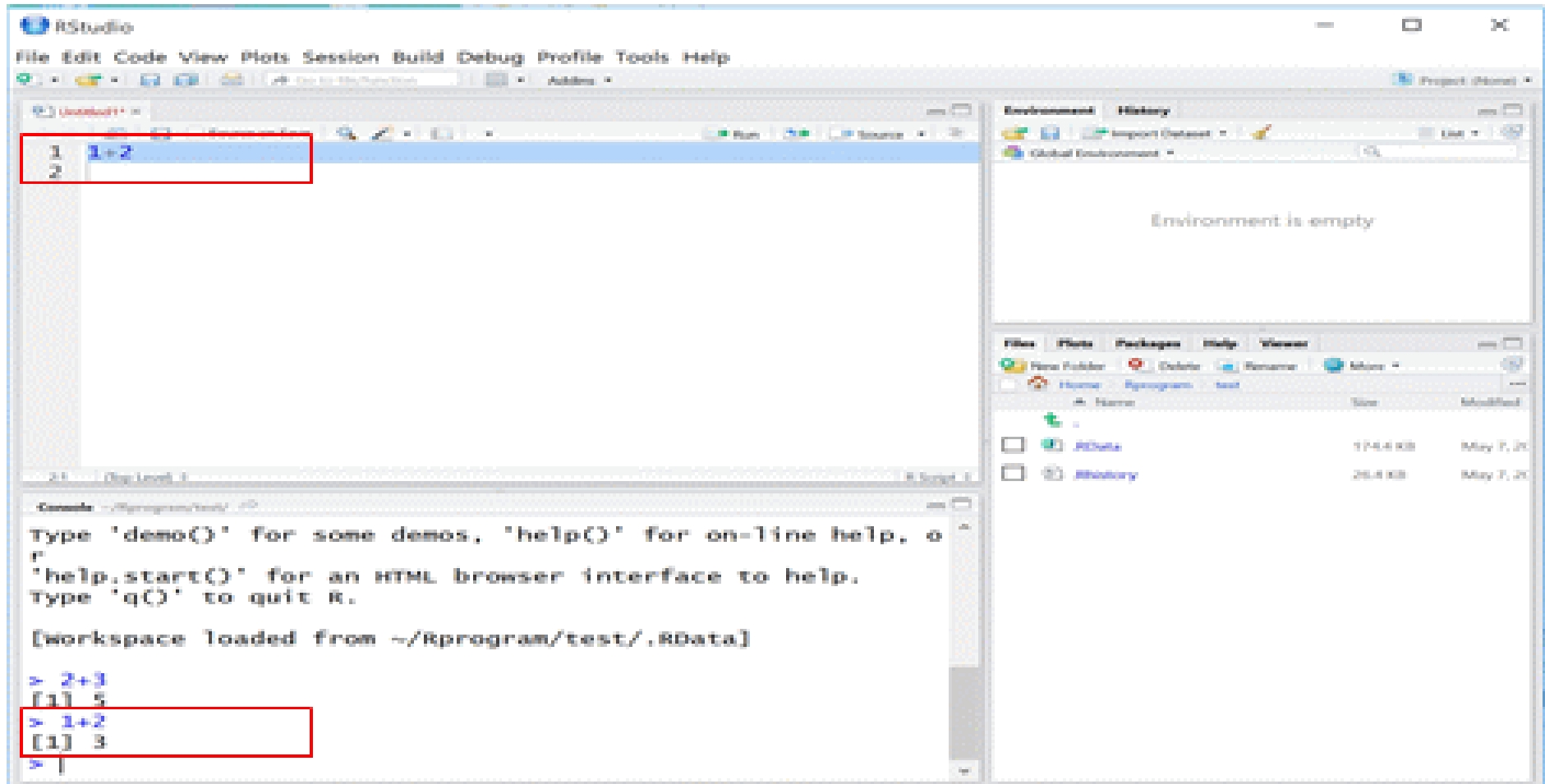
RStudio는 사용자 계정이 한글인 경우 경로를 제대로 인식하지 못하는 문제가 있어 수행 도중 에러가 나는 경우가 종종 있다.



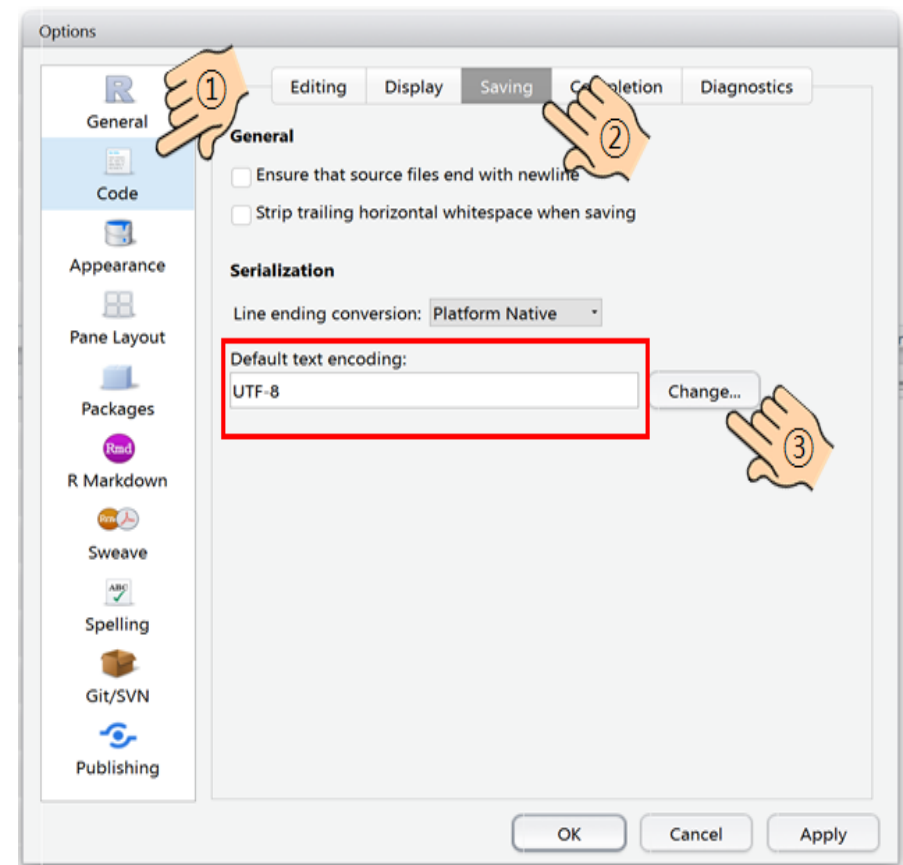
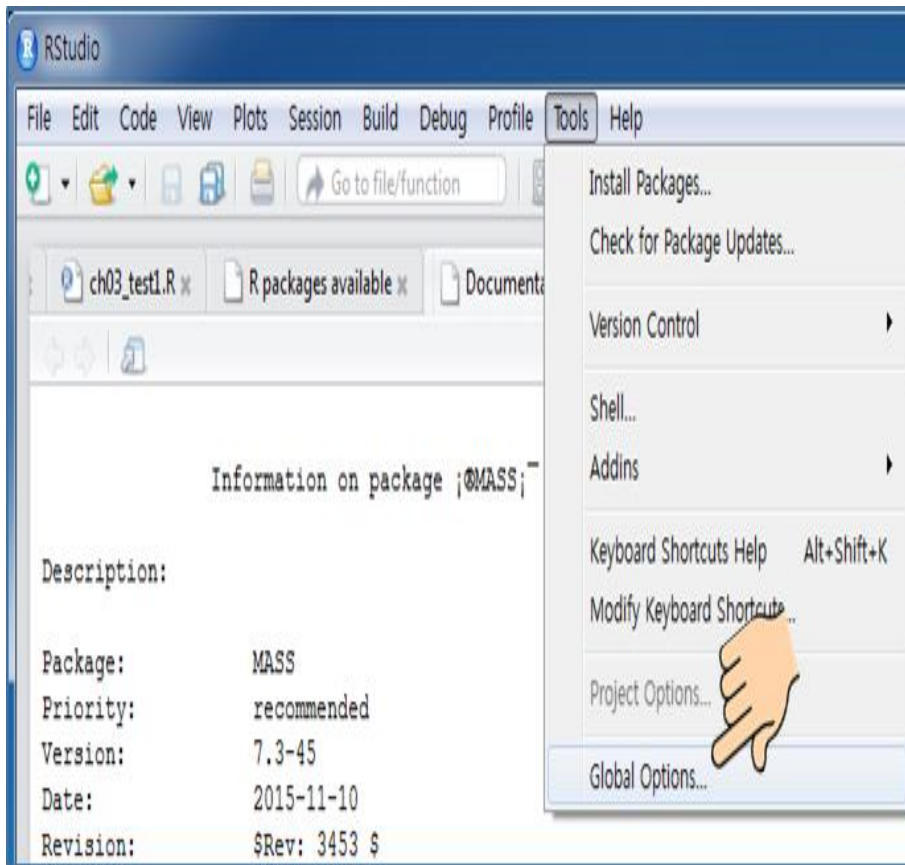
RStudio 화면



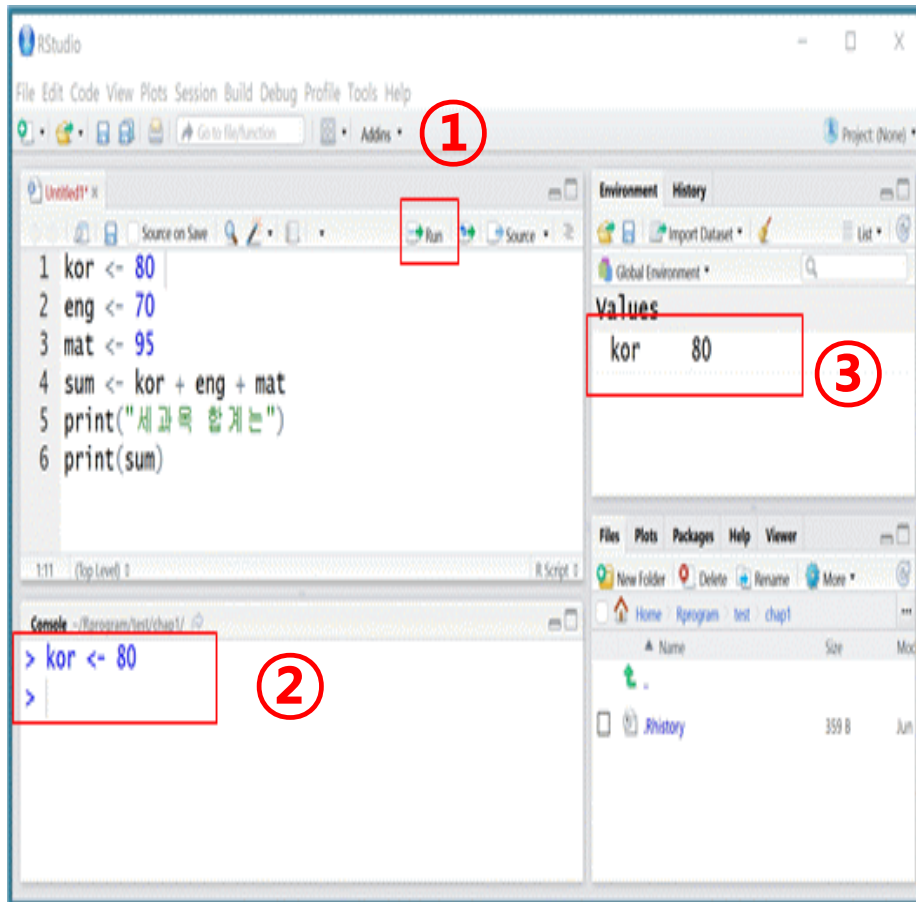
R 명령어 실행



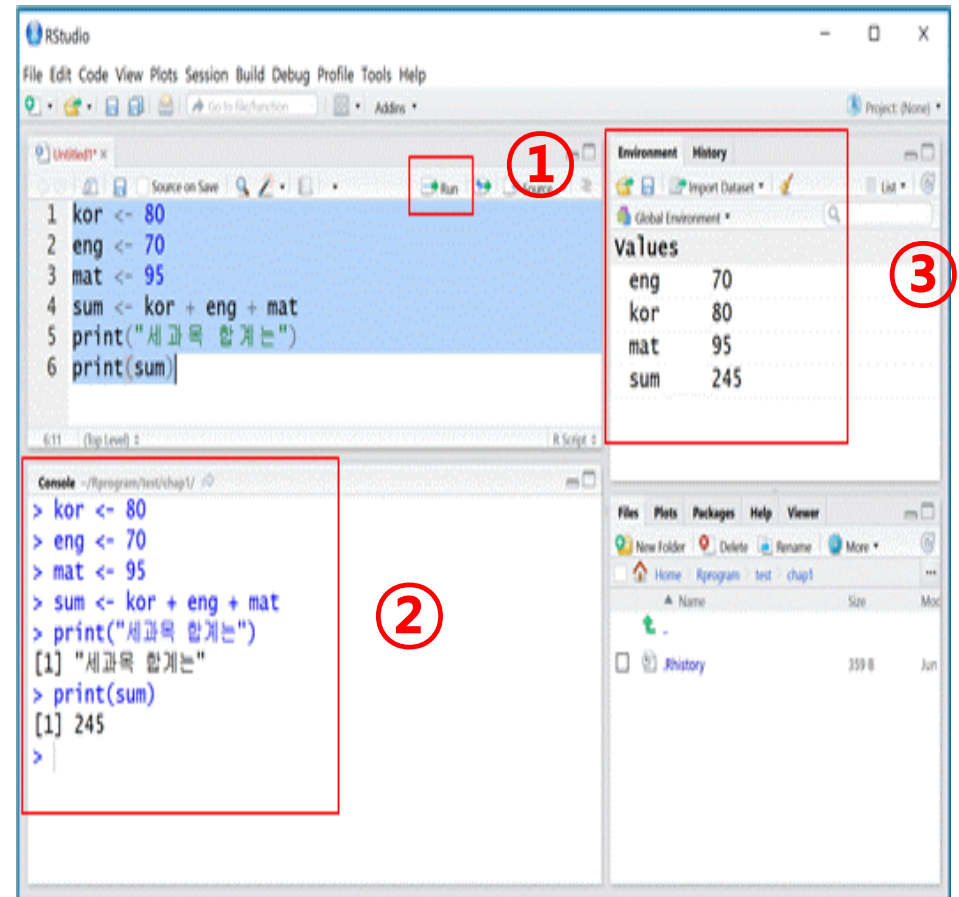
RStudio 환경 설정



R 스크립트 실행하기



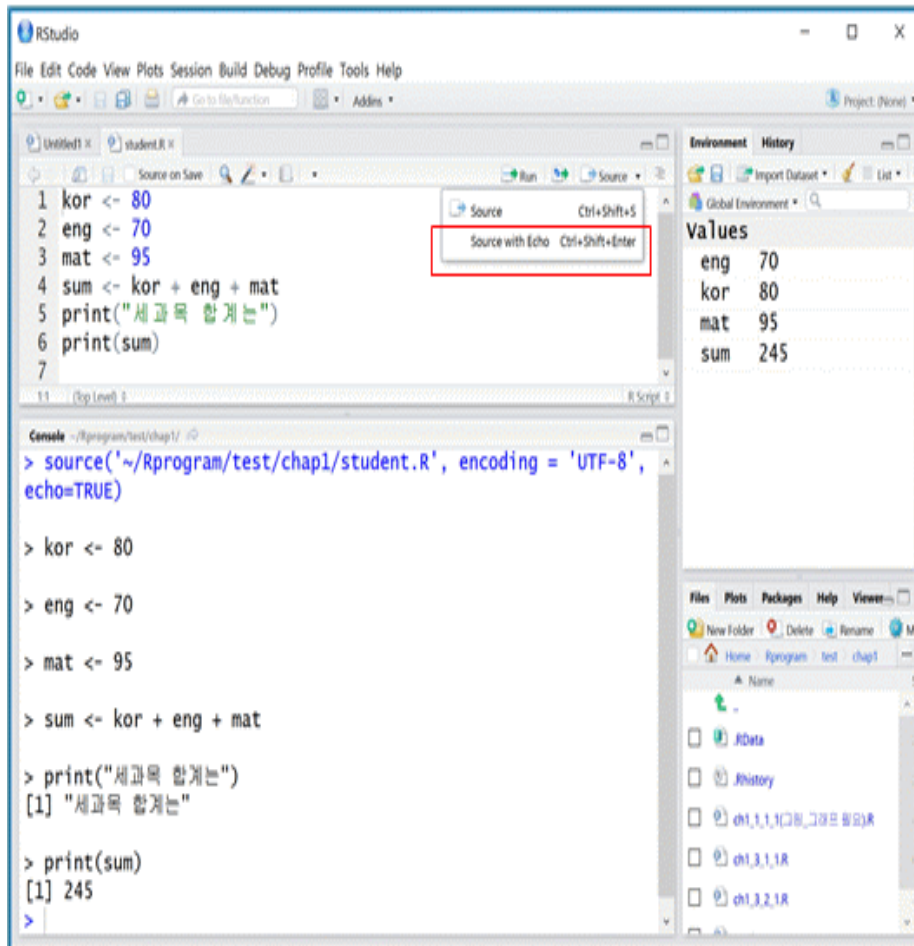
한줄씩 실행하기



블록으로 여러 줄 실행하기

R 프로그램 파일 한 번에 실행하기

Ctrl + Shift + Enter (Source with echo)로 실행한 경우



This screenshot shows the RStudio interface. The script editor on the left contains the following R code:

```
1 kor <- 80
2 eng <- 70
3 mat <- 95
4 sum <- kor + eng + mat
5 print("세과목 합계는")
6 print(sum)
7
```

The 'Source' button in the toolbar is highlighted with a red box, and a tooltip shows 'Source with Echo Ctrl+Shift+Enter'. The Environment pane on the right shows the values of the variables: eng (70), kor (80), mat (95), and sum (245). The Console pane at the bottom shows the output of the code execution:

```
> source('~\Rprogram\test\chap1\student.R', encoding = 'UTF-8',
echo=TRUE)

> kor <- 80

> eng <- 70

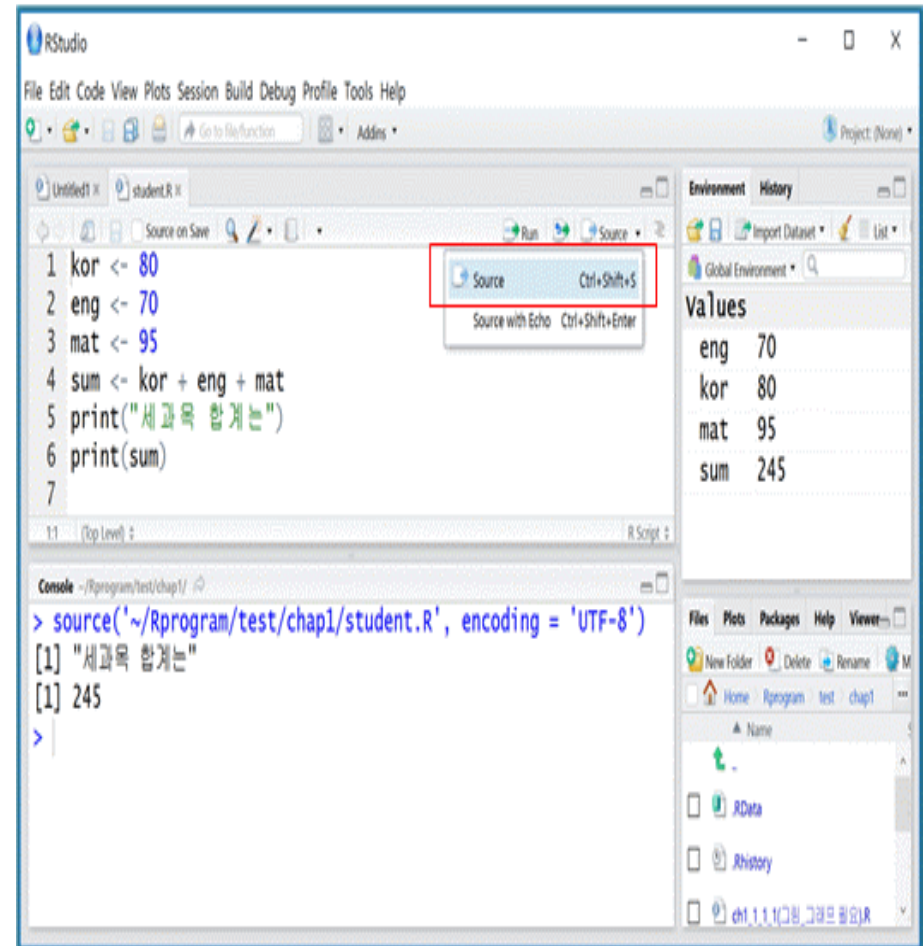
> mat <- 95

> sum <- kor + eng + mat

> print("세과목 합계는")
[1] "세과목 합계는"

> print(sum)
[1] 245
>
```

Ctrl + Shift + S (Source)로 실행한 경우



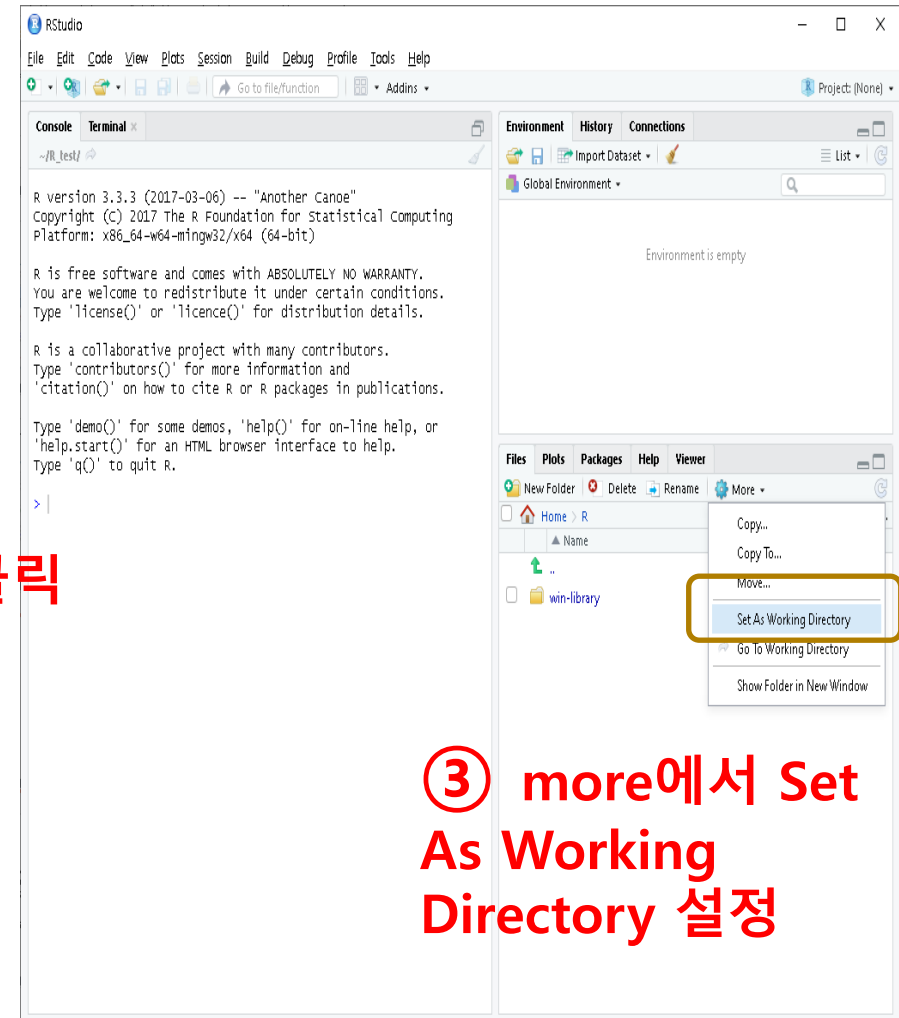
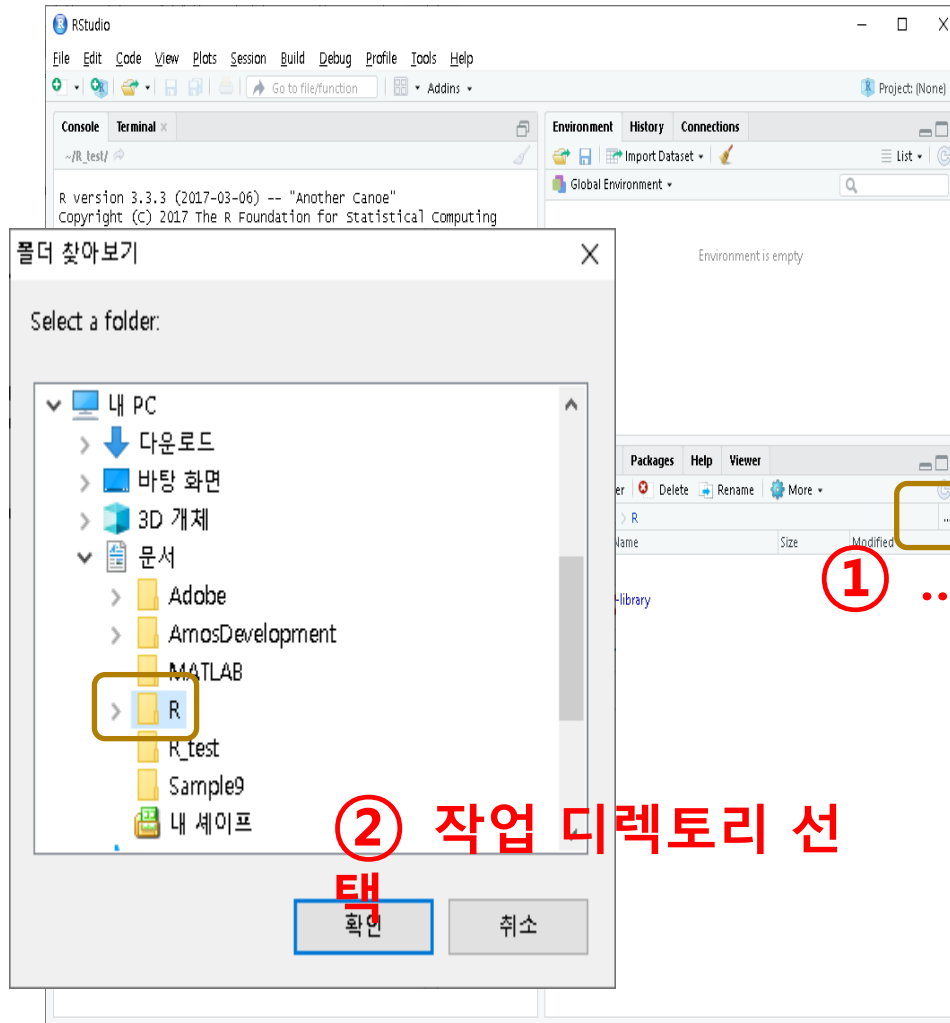
This screenshot shows the RStudio interface. The script editor on the left contains the same R code as the previous screenshot:

```
1 kor <- 80
2 eng <- 70
3 mat <- 95
4 sum <- kor + eng + mat
5 print("세과목 합계는")
6 print(sum)
7
```

The 'Source' button in the toolbar is highlighted with a red box, and a tooltip shows 'Source Ctrl+Shift+S'. The Environment pane on the right shows the same variable values: eng (70), kor (80), mat (95), and sum (245). The Console pane at the bottom shows the output of the code execution:

```
> source('~\Rprogram\test\chap1\student.R', encoding = 'UTF-8')
[1] "세과목 합계는"
[1] 245
>
```


작업용 기본 디렉토리



R 언어는

- R은 1줄씩 처리하는 인터프리터 언어이다
- > 는 명령 프롬프트 입니다. 원하는 명령을 치고 엔터를 치면 실행이 됩니다.
- R은 대소문자를 구분하므로 명령을 실행할 때 주의해야 한다.
- 만약 이전에 했던 작업을 다시 수행하고 싶으면 위로 가는 방향키를 사용한다.
- R 을 종료하려면 q() 를 사용하면 됩니다.
- 작업하는 내용을 저장하거나 작업용 데이터를 보관하는 작업 디렉터리를 지정하고 진행한다.
- # 기호는 주석으로 인식합니다



데이터 불러오기 / 저장하기



파일 불러오기/저장하기 -readLines /writeLines

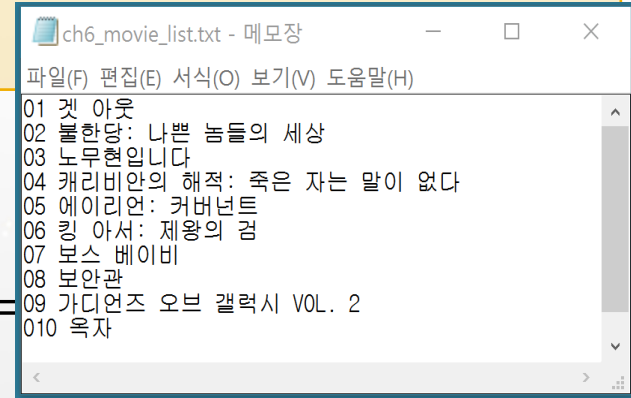
`readLines("읽어올 파일명", n=읽어올 라인수 : 파일 읽어오는 함수`
`writeLines(문자열벡터, "저장할 파일명")`

한 줄 단위로 4줄만 읽어오기

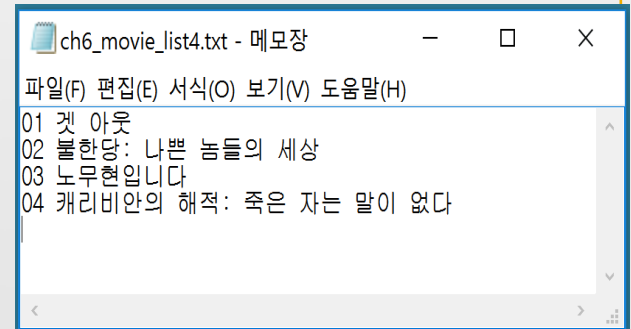
```
txtLine4 <- readLines("트럼프국회연설문.txt", n = 4)
```

읽어온 4줄을 파일에 저장하기

```
writeLines(txtLine4, "트럼프국회연설문-cp.txt")
```



```
ch6_movie_list.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
01 갯 아웃
02 불한당: 나쁜 놈들의 세상
03 노무현입니다
04 캐리비안의 해적: 죽은 자는 말이 없다
05 에이리언: 커버넌트
06 킹 아서: 제왕의 검
07 보스 베이비
08 보안관
09 가디언즈 오브 갤럭시 VOL. 2
10 육자
```



```
ch6_movie_list4.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
01 갯 아웃
02 불한당: 나쁜 놈들의 세상
03 노무현입니다
04 캐리비안의 해적: 죽은 자는 말이 없다
```

파일 불러오기 함수 – read.table

```
read.table("파일명",           # 불러올 파일명
           header = TRUE,      # 첫 행에 열의 이름이 있는지 여부
           sep = ",",          # 데이터 간의 구분자 , 또는 : 등
           skip = 2            # 건너 뛴 줄 수
           nrows = 3           # 읽어올 줄 수
           stringsAsFactor = FALSE # 문자데이터를 요인(Factor)로 다룰 것인지
           ... )               # 아니면 단순 문자열로 다룰 것인지 여부
```

[Script]

```
> score1 <- read.table("ch6_score_space.txt", header = TRUE,
  stringsAsFactors = FALSE)
> print(score1)

> score2 <- read.table("ch6_score_comma.csv", header = TRUE,
  sep = ",", stringsAsFactors = FALSE)
> print(score2)
```

파일 저장하기 함수 – write.table

```
write.table(data,                # 저장할 데이터
            "파일명",           # 저장될 파일명
            row.names = TRUE,    # 행에 1,2,3 자동으로 번호를 붙일지 여부
            col.names = TRUE,    # 열에 자동으로 열이름을 붙일지 여부
            sep = ",",          # 데이터 간의 구분자 , 또는 : 등
            quote = FALSE       # 문자데이터와 행/열 이름에 " "를 붙일지 여부
            ... )
```

[Script]

```
> kor = c(80, 90, 70)
> write.table(kor, "kor1.txt")
> write.table(kor, "kor2.txt", quote=FALSE, row.names = FALSE,
              col.names = FALSE)
```

파일 불러오기 함수 – read.csv

```
read.csv("파일명",  
        header = TRUE,  
        sep = ":",  
        skip = 2 ,  
        nrows = 3 ,  
        stringsAsFactor = FALSE  
        ... )
```

불러올 파일명
첫 행에 열의 이름이 있는지 여부
데이터 간의 구분자
건너 뛴 줄 수
읽어올 줄 수
문자열을 factor로 인식할지 여부

[Script]

```
> # csv 파일에서 데이터 읽어와서 데이터 프레임으로 저장  
> yellowDust <- read.csv("서울시 연도별 황사 경보발령 현황.csv")  
> print(yellowDust)
```

한글이 깨질 경우

```
read.csv("서울시 연도별 황사 경보발령 현황.csv",  
        encoding='utf-8')
```

파일 저장하기 함수 – write.csv

```
write.csv(data,  
  "파일명",  
  row.names = TRUE,  
  col.names = TRUE,  
  sep = ":",  
  quote = FALSE  
  ... )
```

```
# 저장할 데이터  
# 저장될 파일명  
# 행에 1,2,3 자동으로 번호를 붙일지 여부  
# 열에 자동으로 열이름을 붙일지 여부  
# 데이터 간의 구분자 " : " 등  
# 문자데이터와 행/열 이름에 " "를 붙일지 여부
```

[Script]

- > # daysYellowDust를 yellowDust.csv 파일로 저장
- > write.csv(daysYellowDust, "yellowDust.csv", row.names = FALSE,
 quote = TRUE)

엑셀 파일 불러오기 함수 – read.xlsx

<code>read.xlsx("파일명",</code>	<code># 불러올 파일명</code>
<code> sheetIndex</code>	<code># 불러올 시트 번호</code>
<code> startRow = 1,</code>	<code># 불러올 행의 시작</code>
<code> endRow = 10</code>	<code># 불러올 행의 끝</code>
<code> colIndex = c(1,3,..)</code>	<code># 불러올 열 벡터</code>
<code> colClasses = c("numeric",..)</code>	<code># 열의 데이터 유형 변환</code>
<code> header = TRUE</code>	<code># 열이름이 있는지 여부</code>
<code> encoding = "UTF-8"</code>	<code># 문자열 코드</code>
<code> ...)</code>	

[Script]

```
> # "xlsx" 패키지 인스톨 필요
> install.packages("xlsx")
> library(xlsx)
> movie1 <- read.xlsx("주간_주말_박스오피스.xlsx", 1, startRow = 4,
endRow = 21, colIndex = c(1, 2, 4, 9, 12), header = TRUE, encoding
= "UTF-8")
```

엑셀 파일 저장하기 함수 – write.xlsx

```
write.xlsx(data,  
  "파일명",  
  sheetName = "Sheet1",  
  col.names = FALSE,  
  row.names = FALSE,  
  ... )
```

```
# 저장할 데이터  
# 저장될 파일명  
# 저장할 시트 이름  
# 열에 자동으로 열이름을 붙일지 여부  
# 행에 자동으로 행번호를 붙일지 여부
```

[Script]

```
> # 엑셀 파일에 저장하기
```

```
> write.xlsx(movie2, "boxoffice.xlsx", row.names = FALSE)
```

● 파일을 읽고 쓰는 함수 정리

파일 확장자가 .txt이거나 .csv일 경우

(주의할 점.. 파일 확장자가 txt이면 sep=" ", csv이면 sep=";")

```
> data <- read.table("파일명", sep=" ", header=T, skip = 2)
> write.table(data, "파일명", sep=" ", row.names=F )
```

파일 확장자가 .xlsx 인 경우

(주의할 점.. 만약 한글이 포함된 파일일 경우 read.xlsx2 사용)

```
> install.packages("xlsx")
> library("xlsx")

> data <- read.xlsx("파일명.xlsx", sheetIndex = 1, startRow = 2, header=T)
> write.xlsx(data, "파일명.xlsx", sheetName="sheet1", row.names = F)
```




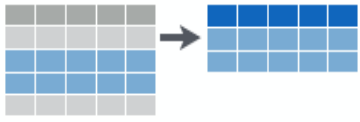


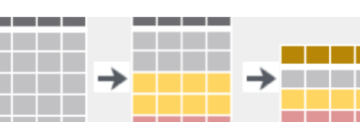

데이터 가공 및 분석



데이터 가공 및 분석 - dplyr 패키지

- 데이터분석을 위한 전처리
 - "자신이 알고자하는 방향"에 맞게 데이터를 추출하고, 나누는 과정

dplyr 패키지의 주요 함수

함수		예
select(선택할것) select(-제외할것)	열 선택	<pre>train %>% select(Survived)</pre> <pre>train %>% select(-PassengerId)</pre> 
filter(데이터, 기준)	행 추출	<pre>filter(train, is.na(Age)==TRUE)</pre> 
mutate(데이터, 새변수 = 내용)	열 추가	<pre>mutate(train, family = 1+ifelse(SibSp == 1 , 1 ,0)+Parch)</pre> 
arrange(데이터, 오름차순기준) arrange(데이터, 내림차순기준)	행 정렬	<pre>arrange(train, Age)</pre> <pre>arrange(train, desc(Age))</pre> 
group_by(그룹핑할 기준)	행 결합	<pre>train %>% group_by(sex)</pre> 
summarise (그룹핑 된 결과로 만들 내용)	행 요약	<pre>train %>%</pre> <pre> group_by(sex) %>%</pre> <pre> summarise(n())</pre> <p>#n()은 갯수를 세는 방법입니다.</p> 

dplyr Example(1)

```
install.packages("dplyr")      #패키지 설치
```

```
library(dplyr)                #메모리 로드
```

```
score <- read.csv("score.csv")
```

```
str(score )
```

```
head(arrange(score ,class))    #반별로 5개만 조회
```

```
tail(arrange(score, desc(class)), 5)    #반별 내림차순으로 하위5개 조회
```

```
arrange(score, desc(class)) %>% tail(5)
```



파이프 연산자는 **%>%** 첫번째 함수의 출력을 두번째 함수의 입력으로 바꾸어 준다

dplyr Example(2)

#2반 학생의 점수 만 조회

```
filter(score, class==2) %>% head()
```



#중간고사 점수가 70점 이상인 학생 조회

```
filter(score, mid >= 70) %>% head()
```

중간 , 기말 모두 70점 이상인 학생은??

```
filter(score, mid >= 70 & final >= 70) %>% head()
```

반, 중간고사와 기말 고사 필드만 조회

```
select(score, class, mid , final) %>% View()
```

2반 학생 중 중간고사 70점 이상인 학생의 중간, 기말 고사 성적을 조회 ??

연산자	의미
<	작다
>	크다
<=	작거나 같다
>=	크거나 같다
==	같다
!=	같지않다

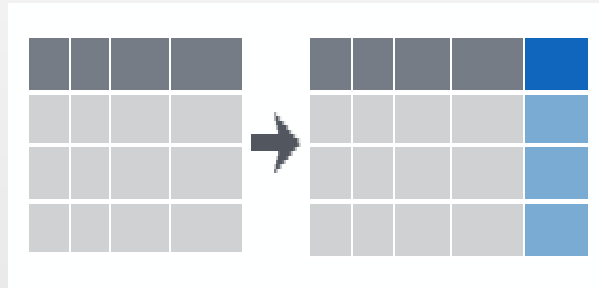
연산자	의미
&	논리곱(and)
	논리합(or)

dplyr Example(3)

#평균을 계산해서 추가해 보자

#평균점수는 중간 40 기말 40 레포트 10 출석 10

```
score <- mutate(score, avg=(mid*0.4 + final*0.4 + report*0.1 +  
attend*0.1) )
```



dplyr Example(4)

반별 최고 평균점수와 최저 평균점수를 알아보자

```
score %>% group_by(class) %>%
```

```
  summarise(avg_max = max(avg), avg_min = min(avg) )
```

- summarise()에서 자주 사용하는 요약 통계량 함수

함수	의미
mean()	평균
sd()	표준편차
sum()	합계
median()	중앙값
min()	최솟값
max()	최댓값
n()	빈도



Mpg 데이터를 이용해 문제를 해결해 보자(1)

- 배기량에 따라 연비는 어떻게 달라지는가?
 - 자동차의 배기량이 4이하인 차들과 배기량이 5이상인 차들의 고속도로 연비의 평균을 비교해보자
 - 풀이))
 - 배기량(displ)이 4이하인 조건에 해당하는 데이터를 추출해서 새로운 변수에 할당한 다음 평균을 한다.
 - 배기량(displ)이 5이상인 조건에 해당하는 데이터를 추출해서 새로운 변수에 할당한 다음 평균을 한다.
- 제조회사에 따라 도시 연비가 다른지 알아보자
 - audi와 toyota 중 도시연비의 평균을 구해 비교해보자

Mpg 데이터를 이용해 문제를 해결해 보자(2)

- audi 에서 생산한 자동차 모델 중 고속도로연비(hwy)가 높은 모델을 1위에서 5위까지 알아보자
 - audi 생산 모델을 추출한다
 - 고속도로연비의 내림차순으로 정렬한다
 - 5개만 출력
- 평균연비가 가장 높은 자동차 3종을 출력하시오.
 - 통합연비 변수를 만들어 분석하자
 - 풀이)
 - cty와 hwy를 합한 값을 total변수를 추가하여 mpgC데이터에 저장한다
 - 통합연비의 내림차 순으로 정렬한다.
 - 3개만 출력한다

Mpg 데이터를 이용해 문제를 해결해 보자(3)

- 자동차의 클래스별 어떤 차종의 도시 연비가 높은지 비교해 보려고 한다.
 - 클래스별 도시연비 평균을 구해보자
 - 도시연비 평균이 높은 순으로 정렬해 보자
- Quiz
 - 회사별로 "suv" 자동차의 도시 및 고속도로의 통합연비 평균을 구해 내림차순으로 정렬하고, 1위부터 5위까지 출력해보자

Report

- * 각 팀별로 각 학과에 맞는 공공데이터를 다운로드 받으시오
- * 다운로드 받은 데이터에서 필요한 데이터를 조회할 수 있도록 질의문을 3개 이상 만드시오.
- * dplyr 패키지의 여러 함수들을 사용하여 질의문을 해결하시오.

GitHub 깃허브



프로그래머를 위한 소셜 코딩 공간- 깃허브

- “당신은 깃허브에 대해 잘 모를 수도 있지만, 소프트웨어 개발자 사이에서 깃허브는 메카(최고의 성지)다.”

포춘이 설명한 깃허브에 대한 묘사

깃허브는 오픈소스 소프트웨어의 중심지(hub) 역할을 하면서 오픈소스 프로젝트가 널리 퍼지는 데 크게 기여하고 있다.



GIT

- 깃은 2005년에 개발된 분산형 버전관리 시스템(DVCS)
- 오픈소스 소프트웨어
- 리눅스를 만든 리누스 토발즈와 주니오 하마노가 개발
- 어떤 코드를 수정했는지 기록하고 추적할 수 있다
- 많은 사람들이 함께 소프트웨어를 개발할 때 유용하다.
- 관리자는 여러 사람의 코드 중 일부를 합쳐가며 완성본을 만들어갈 수 있다.
- 수천명의 사람들이 이용해도 안정적이며 중앙저장소에 의존하지 않아 속도도 훨씬 빠르다는 장점이 있다.
- 2018년 6월 MS가 깃허브 인수

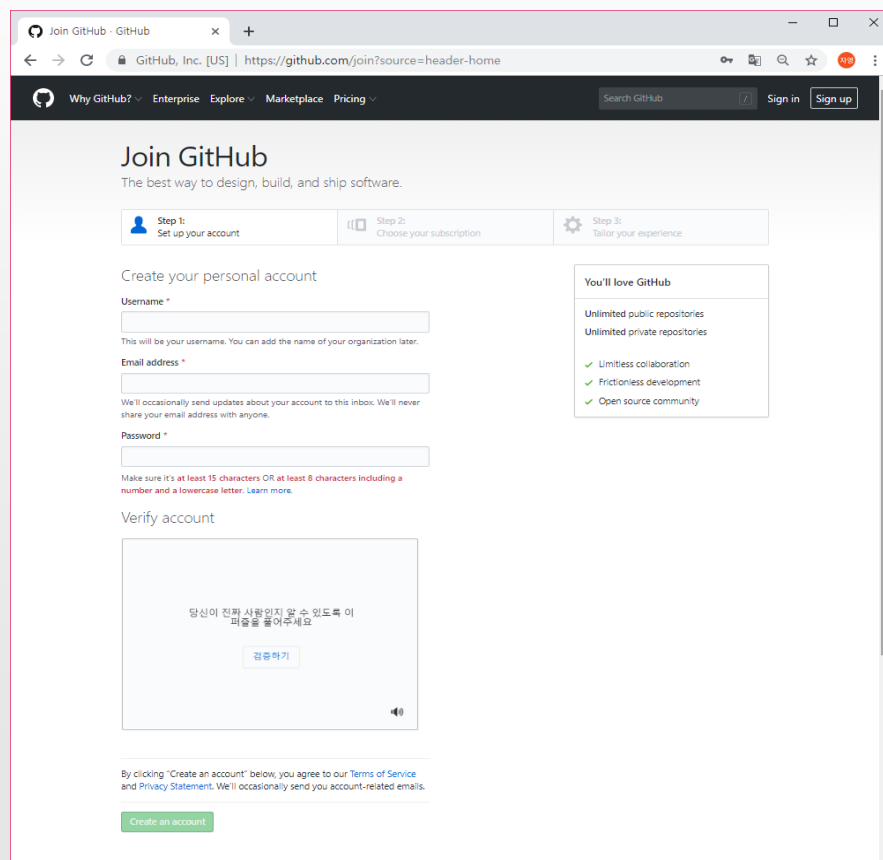
깃허브

- 가장 강력한 기능은 '포크'다
 - 포크는 내 계정으로 외부 프로젝트 코드 저장소를 그대로 복사해주는 것
- 깃허브 덕에 소스코드를 복사하고 배포하던 과정을 클릭 몇 번만으로 가능
- 웹사이트에 소스코드가 공개되니 검색도 용이
- 개발자들만의 강력한 커뮤니티
 - 프로필 작성
 - 프로젝트와 코드 공개
 - 구직 활동에 활용
- 구글, 페북, MS, 애플 등 많은 IT 기업들이 오픈소스 프로젝트를 깃허브에서 관리
- 프라이빗 저장소는 유료, 기업용 깃허브 유료

출처 : 네이버 지식백과

깃허브 회원가입(1)

- <https://github.com/> 접속
 - [Microsoft Edge](#), [Google Chrome](#), or [Firefox](#).
- [Sign up] 클릭
- Username
 - Username.github.io
- Email Address
- Password
- 입력 후
- Verify account 확인
- Create Account 클릭



Join GitHub · GitHub

GitHub, Inc. [US] | <https://github.com/join?source=header-home>

Why GitHub? Enterprise Explore Marketplace Pricing

Search GitHub / Sign in Sign up

Join GitHub

The best way to design, build, and ship software.

Step 1: Set up your account Step 2: Choose your subscription Step 3: Tailor your experience

Create your personal account

Username *

This will be your username. You can add the name of your organization later.

Email address *

We'll occasionally send updates about your account to this inbox. We'll never share your email address with anyone.

Password *

Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more](#).

Verify account

당신이 진짜 사는지 알 수 있도록 이 퍼즐을 풀어주세요

검증하기

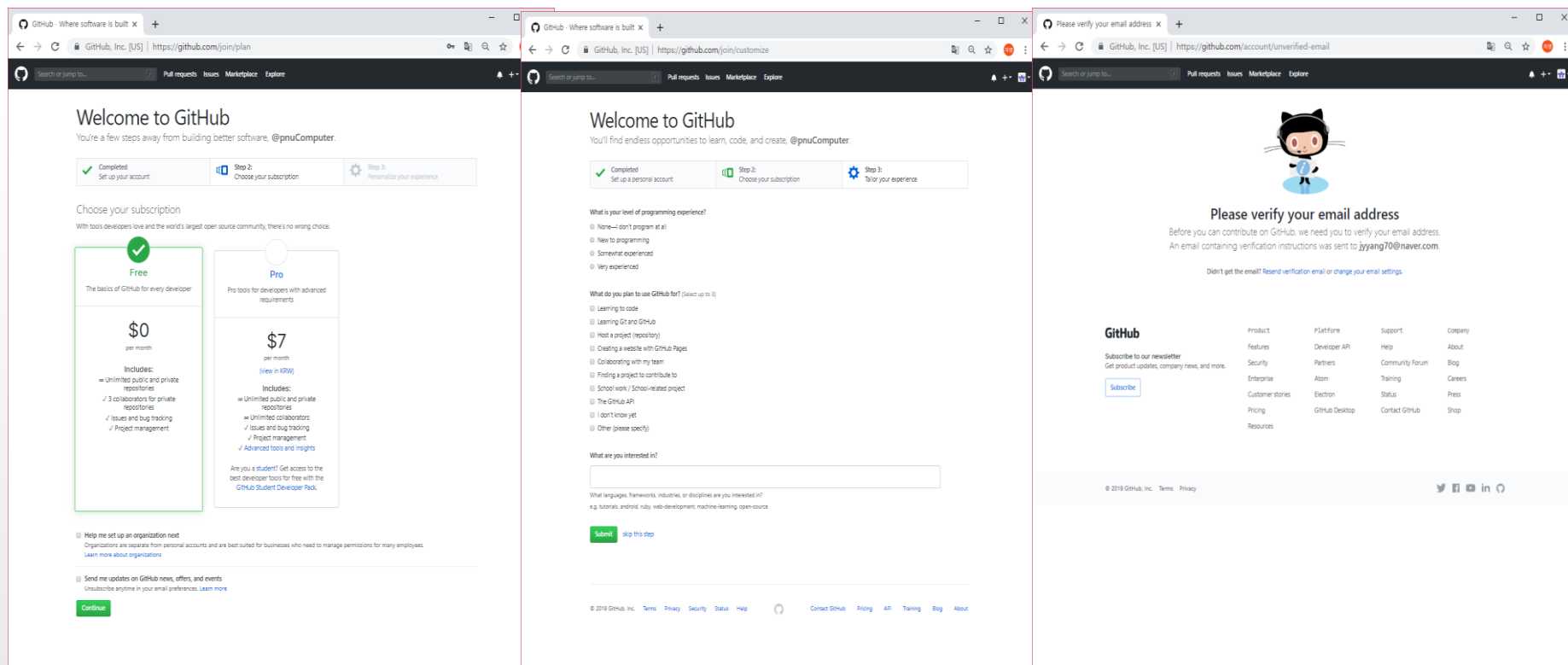
By clicking "Create an account" below, you agree to our [Terms of Service](#) and [Privacy Statement](#). We'll occasionally send you account-related emails.

Create an account

You'll love GitHub

- Unlimited public repositories
- Unlimited private repositories
- ✓ Limitless collaboration
- ✓ Frictionless development
- ✓ Open source community

깃허브 회원가입(2)



The first screenshot shows the 'Welcome to GitHub' page with a progress bar indicating three steps: 'Completed: Set up a personal account', 'Step 2: Choose your subscription', and 'Step 3: Personalize your experience'. Under 'Choose your subscription', there are two options: 'Free' (The basics of GitHub for every developer, \$0 per month) and 'Pro' (Pro tools for developers with advanced requirements, \$7 per month). The 'Free' option includes unlimited public and private repositories, 3 collaborators for private repositories, issues and bug tracking, and project management. The 'Pro' option includes unlimited public and private repositories, unlimited collaborators, issues and bug tracking, project management, and advanced tools and insights. There is also a note for students to get access to the best developer tools for free with the GitHub Student Developer Pack. At the bottom, there are checkboxes for 'Help me set up an organization next' and 'Send me updates on GitHub news, offers, and events'.

The second screenshot shows the 'Welcome to GitHub' page with a progress bar indicating three steps: 'Completed: Set up a personal account', 'Step 2: Choose your subscription', and 'Step 3: Personalize your experience'. Under 'What is your level of programming experience?', there are four radio button options: 'None—I don't program at all', 'New to programming', 'Somewhat experienced', and 'Very experienced'. Under 'What do you plan to use GitHub for?', there is a list of checkboxes for various use cases. At the bottom, there is a 'Submit' button and a 'Skip this step' link.

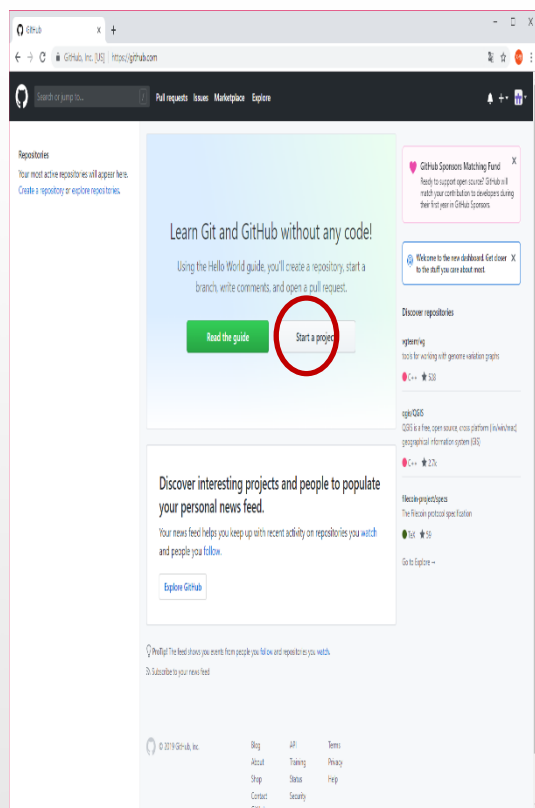
The third screenshot shows the 'Please verify your email address' page. It features the GitHub logo and a message: 'Before you can contribute on GitHub, we need you to verify your email address. An email containing verification instructions was sent to jyyang70@naver.com.' There is a link to 'Didn't get the email? Resend verification email or change your email settings.' At the bottom, there is a footer with copyright information and social media links.

가격 선택

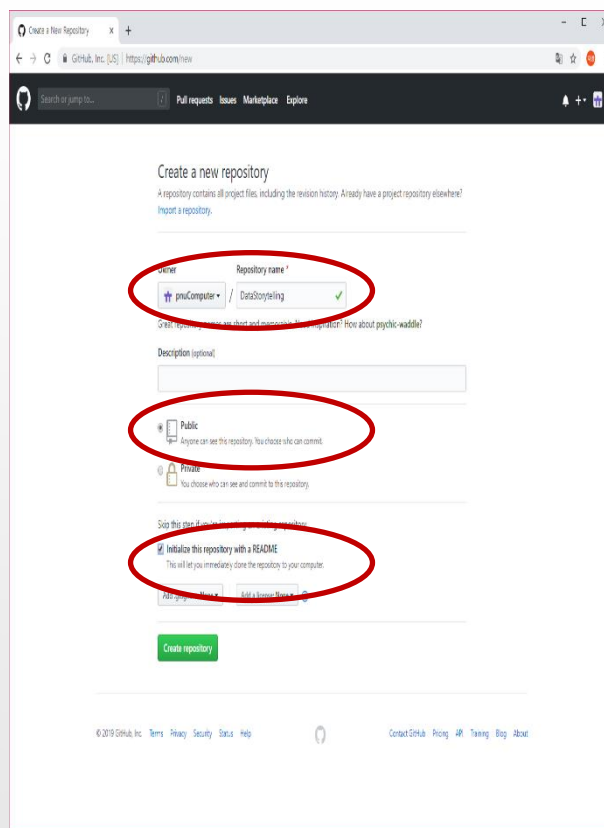
관심분야 체크

완료 후 이메일 인증

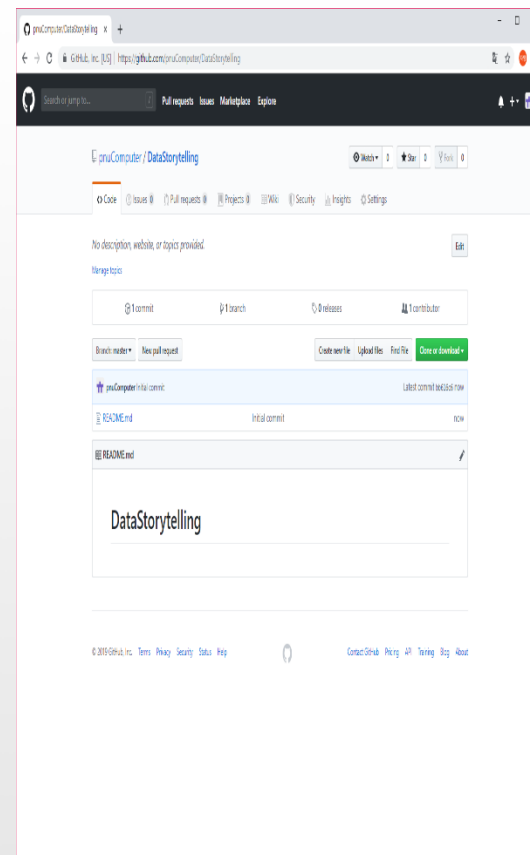
저장소 만들기



Start Project 클릭



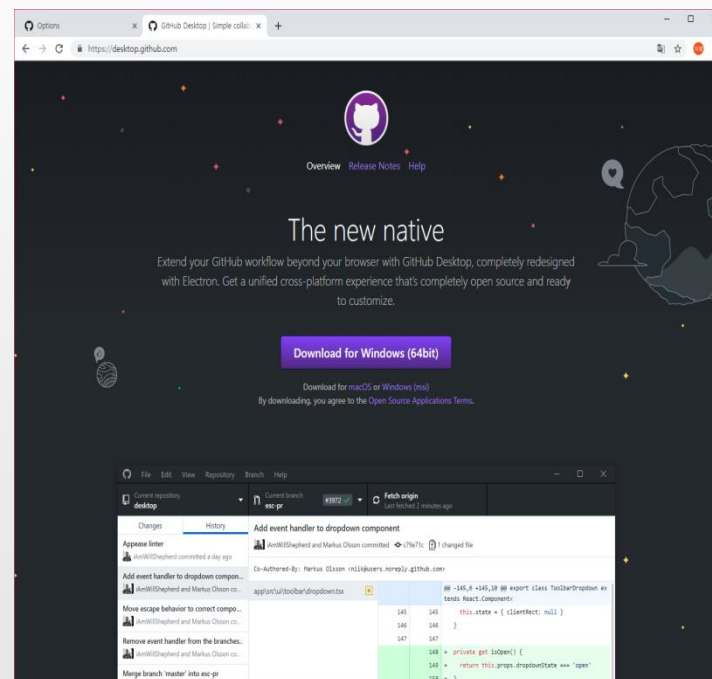
저장소 이름 입력



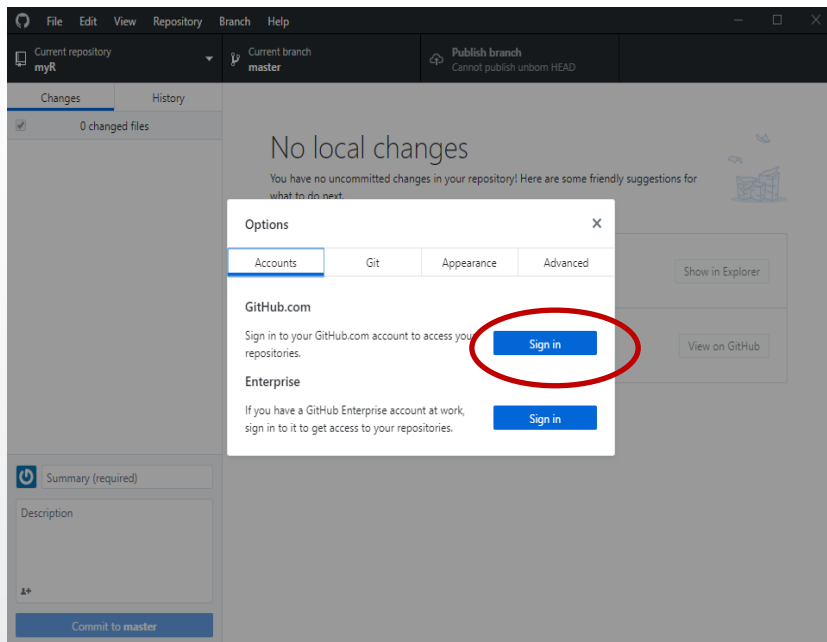
저장소 접근 가능

깃허브 Desktop 설치

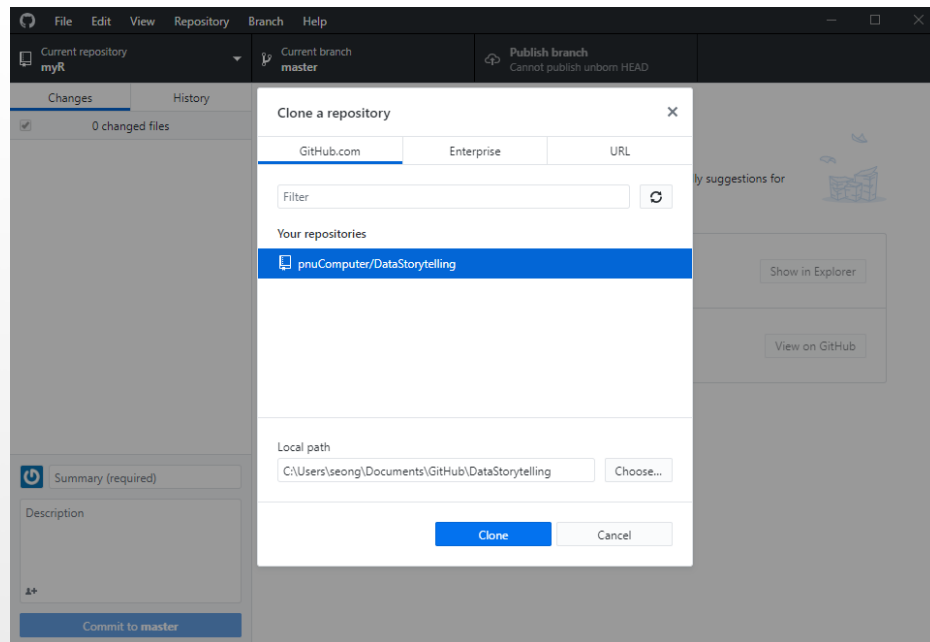
- 로컬컴퓨터에서 Git을 사용할 수 있고 필요할 때 GitHub로 전송 가능
- <https://desktop.github.com/>
- Download for windows
다운로드 및 설치하기



깃허브 Desktop 사용

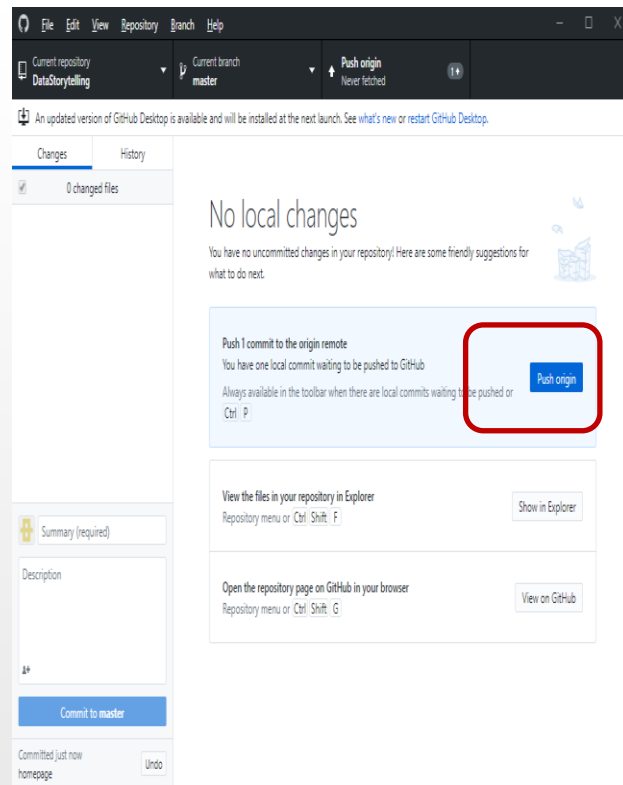
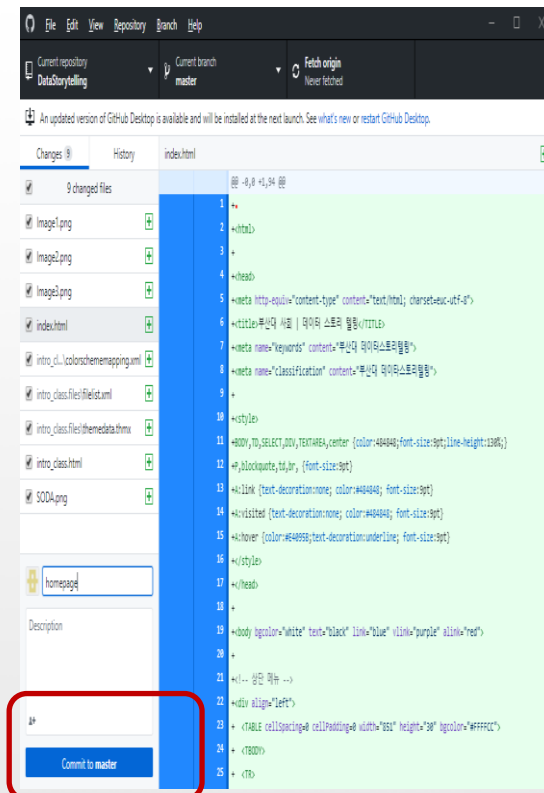
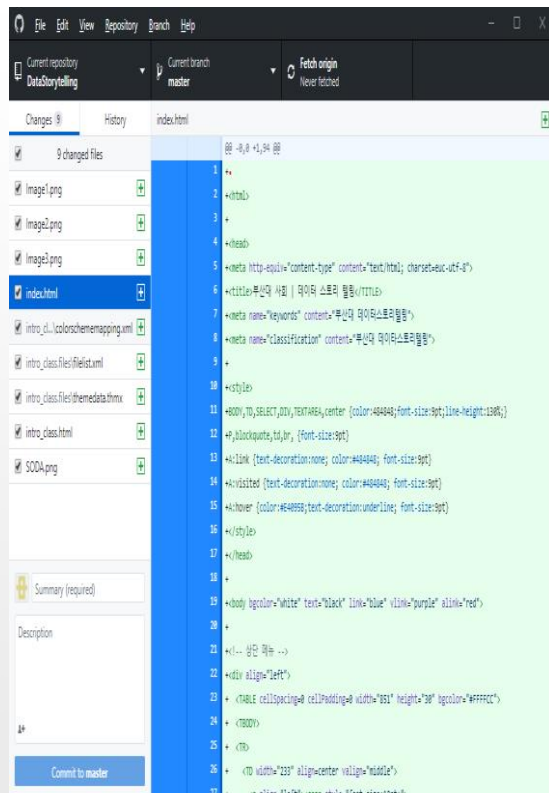


Sing In (로그인)



저장소 Clone(복제하기)

Desktop -> GitHub 전송하기



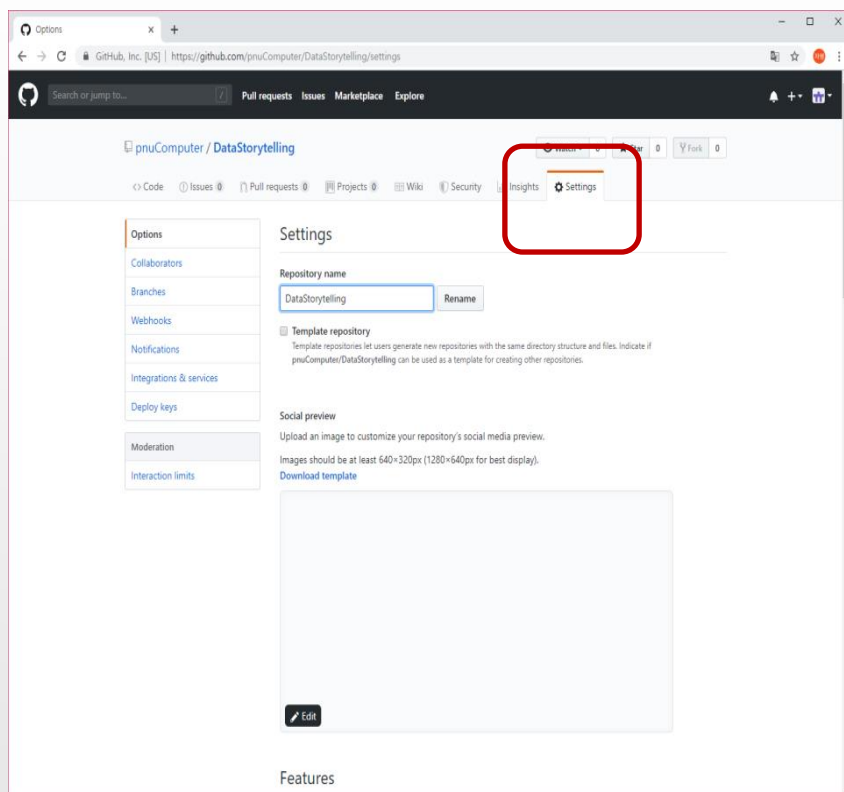
전송할 데이터 선택하기

Desc 적고 Commit 클릭

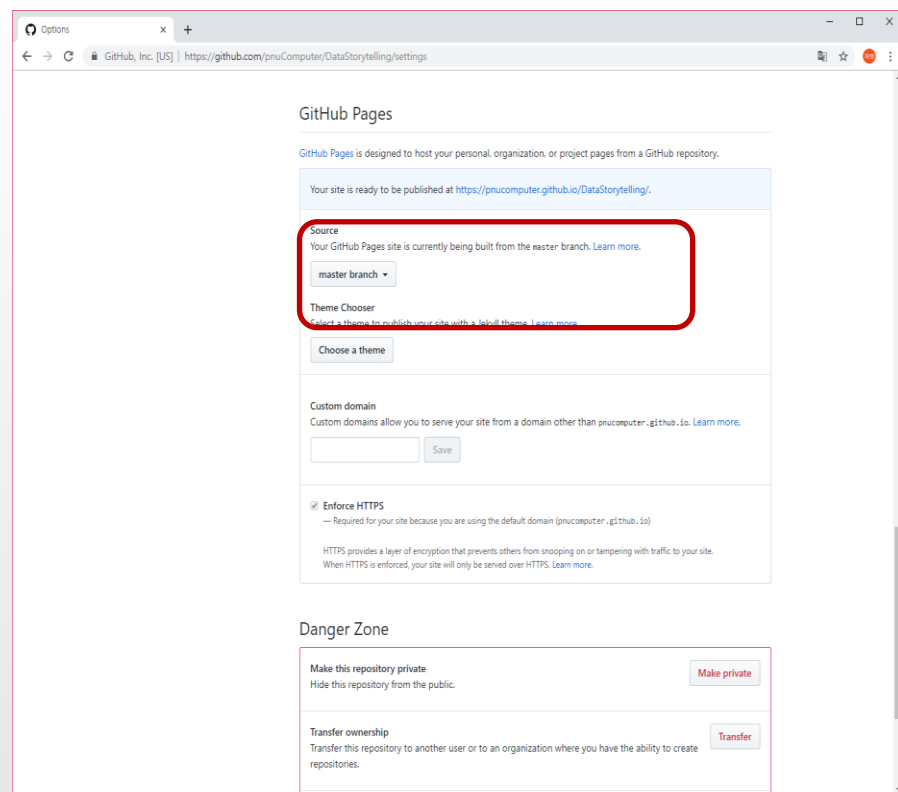
Push origin 클릭하기

깃허브를 블로그로 사용하기(1)

- 웹 주소 만들기

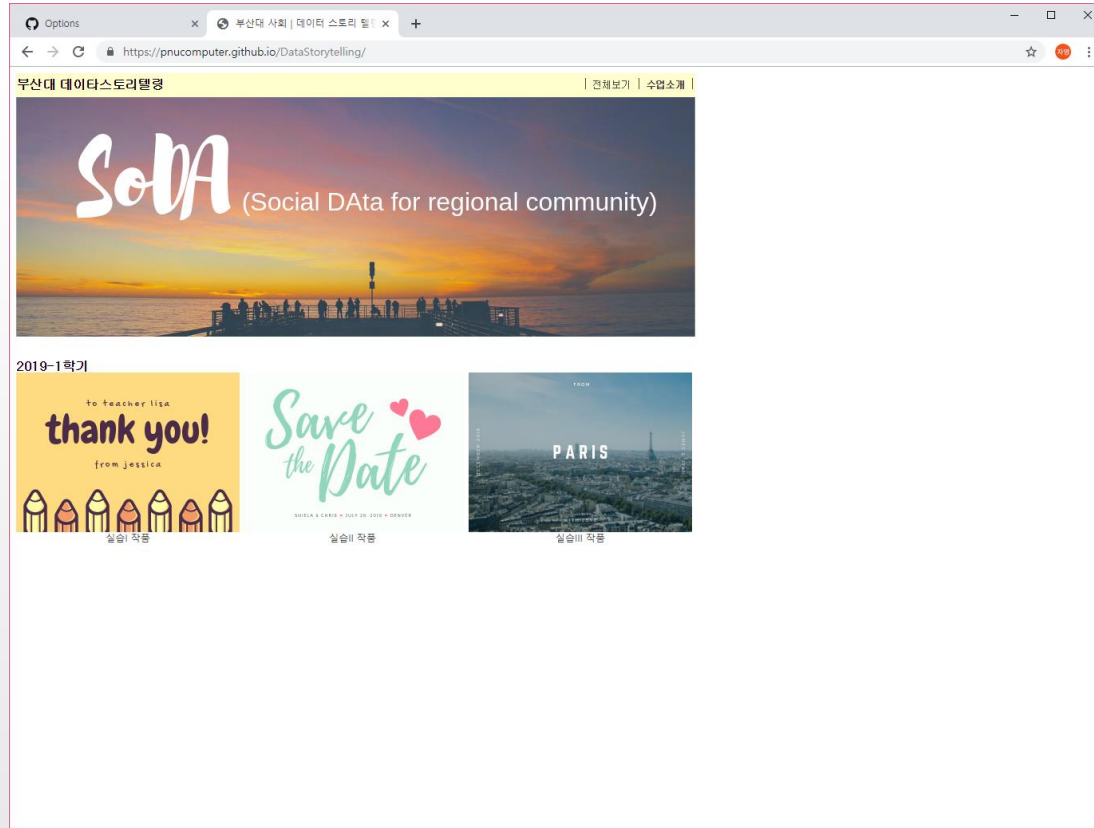


저장소의 Settings 선택하기



GitHub pages 에서
Source 속성을 master branch로 변경

깃허브를 블로그로 사용하기(2)



Index.html을 작성후 GitHub로 전송한 다음
웹에서 주소로 확인한다.