

The background features a large, flowing, green ribbon-like shape that curves across the frame. It has a 3D effect with gradients of green and yellow, giving it a sense of movement. The shape starts from the left, arches over the top, and then curves down towards the bottom right corner. A solid dark green horizontal bar runs along the very bottom of the image.

Load External Data

파일 목록 보기

- `list.files()` 함수 사용해서 파일 목록 표시

```
> list.files()
[1] "data-files"    "lastsave.txt"

> list.files(recursive = T) # 하위 디렉터리 파일도 표시
[1] "data-files/test1.csv"
[2] "data-files/test2.csv"
[3] "data-files/test3.csv"
[4] "lastsave.txt"

> list.files(all.files = T) #숨김 파일도 표시
[1] "."              ".."              ".RData"          ".Rhistory"
[5] "data-files"    "lastsave.txt"
```

텍스트 파일 읽기

- `scan()` 함수를 사용해서 파일 데이터를 읽고 벡터 또는 리스트로 변환

```
> scan1 <- scan("data-files/scan-1.txt")
Read 4 items
> scan1
[1] 111 222 333 444
> scan2 <- scan("data-files/scan-2.txt")
Read 4 items
> scan2
[1] 1 2 3 4
> # what = "" 지정해서 실수, 문자형 처리
> scan2 <- scan("data-files/scan-2.txt", what = "")
Read 4 items
> scan2
[1] "1.00" "2.00" "3.00" "4.00"
```

텍스트 파일 읽기

- `scan()` 함수 사용해서 파일 데이터를 읽고 벡터 또는 리스트로 변환 (계속)

```
> scan3 <- scan("data-files/scan-3.txt", what = "")
Read 4 items
> scan3
[1] "aaa" "bbb" "ccc" "ddd"
> scan3 <- scan("data-files/scan-3.txt")
Error in scan(file, what, nmax, sep, dec, quote, skip, nlines,
na.strings, :
  scan()은 'a real'를 입력받아야 하는데, 'aaa'를 받았습니다
> scan4 <- scan("data-files/scan-4.txt", what = "")
Read 4 items
> scan4
[1] "aaa" "bbb" "111" "2.34"
```

명령행에서 사용자 입력

- 파일 경로 전달인자 없이 `scan()` 함수 사용

```
> input <- scan()
1: 1
2: 2
3: 3
4:   # 입력 없이 엔터 치면 입력 종료
Read 3 items
> input
[1] 1 2 3
> input2 <- scan(what = "") # 문자 입력 받을 때 what = ""
1: a
2: b
3:
Read 2 items
> input2
[1] "a" "b"
```

명령행에서 사용자 입력

- `readline()` 함수로 한 줄 읽기

```
> input3 <- readline()  
Hello, R Programming World !!!  
  
> input3  
[1] "Hello, R Programming World !!!"  
  
> input4 <- readline("Input your name : ")  
Input your name : John Doe  
  
> input4  
[1] "John Doe"
```

텍스트 파일 읽기

- 파일을 한 줄씩 읽어서 벡터에 담기 → `readLines()` 함수 사용

```
> input5 <- readLines("data-files/scan-4.txt")  
> input5  
[1] "aaa"  "bbb"  "111"  "2.34"
```

텍스트 파일 읽기

- `read.table()` 함수로 텍스트 파일을 읽어서 데이터 프레임에 저장

```
> fruits <- read.table("data-files/fruits.txt"); fruits;
```

	V1	V2	V3	V4
1	no	name	price	qty
2	1	apple	500	5
3	2	banana	200	2
4	3	peach	200	7
5	4	berry	50	9

```
> fruits <- read.table("data-files/fruits.txt", header = T); fruits;
```

	no	name	price	qty
1	1	apple	500	5
2	2	banana	200	2
3	3	peach	200	7
4	4	berry	50	9

텍스트 파일 읽기

- `read.table()` 함수로 텍스트 파일을 읽어서 데이터 프레임에 저장 (계속)

```
> fruit2 <- read.table("data-files/fruits-2.txt")
```

```
> fruit2
```

	V1	V2	V3	V4
1	1	apple	500	6
2	2	banana	200	2
3	3	peach	200	7
4	4	berry	50	9

```
> fruit2 <- read.table("data-files/fruits-2.txt", skip = 2)
```

```
> fruit2
```

	V1	V2	V3	V4
1	2	banana	200	2
2	3	peach	200	7
3	4	berry	50	9

텍스트 파일 읽기

- `read.table()` 함수로 텍스트 파일을 읽어서 데이터 프레임에 저장 (계속)

```
> fruit2 <- read.table("data-files/fruits-2.txt", nrows = 2); fruit2;
  V1      V2  V3 V4
1  1  apple 500  6
2  2 banana 200  2

> fruits3 <- read.table("data-files/fruits.txt", header = T, nrows = 2)
> fruits3
  no  name price qty
1  1  apple   500   5
2  2 banana   200   2

> fruits3 <- read.table("data-files/fruits.txt",
                        header = F, skip = 2, nrows = 2)

> fruits3
  V1      V2  V3 V4
1  2 banana 200  2
2  3  peach 200  7
```

CSV 파일 읽기

- `read.csv()` 함수로 csv 파일을 읽어서 데이터프레임에 저장

```
> fruit3 <- read.csv("data-files/fruits-3.csv")
```

```
> fruit3
```

	no	name	price	qty
1	1	apple	500	6
2	2	banana	200	2
3	3	peach	200	7
4	4	berry	50	9

```
> fruit4 <- read.csv("data-files/fruits-4.csv")
```

```
> fruit4
```

	X1	apple	X500	X6
1	2	banana	200	2
2	3	peach	200	7
3	4	berry	50	9

CSV 파일 읽기

- `read.csv()` 함수로 csv 파일을 읽어서 데이터프레임에 저장 (계속)

```
> fruit4 <- read.csv("data-files\\fruits-4.csv", header = F); fruit4;
  V1      V2  V3 V4
1  1  apple 500  6
2  2 banana 200  2
3  3  peach 200  7
4  4  berry  50  9

> label <- c("NO", "NAME", "PRICE", "QTY")
> fruit4 <- read.csv("data-files\\fruits-4.csv", header = F,
                     col.names = label)

> fruit4
  NO  NAME PRICE QTY
1  1  apple  500   6
2  2 banana  200   2
3  3  peach  200   7
4  4  berry   50   9
```

SQL 쿼리를 사용해서 파일 읽기

▪ sqldf 패키지 설치

```
> install.packages("sqldf")
.....
> library(sqldf)
> install.packages("googleVis")
.....
> library(googleVis)
```

SQL 쿼리를 사용해서 파일 읽기

- 실습 데이터 준비

- googleviz 패키지의 Fruits 데이터를 CSV 파일에 저장

```
> Fruits
```

	Fruit	Year	Location	Sales	Expenses	Profit	Date
1	Apples	2008	West	98	78	20	2008-12-31
2	Apples	2009	West	111	79	32	2009-12-31
3	Apples	2010	West	89	76	13	2010-12-31
4	Oranges	2008	East	96	81	15	2008-12-31
5	Bananas	2008	East	85	76	9	2008-12-31
6	Oranges	2009	East	93	80	13	2009-12-31
7	Bananas	2009	East	94	78	16	2009-12-31
8	Oranges	2010	East	98	91	7	2010-12-31
9	Bananas	2010	East	81	71	10	2010-12-31

```
> write.csv(Fruits, "data-files/fruits-sql.csv", quote = F)
```

SQL 쿼리를 사용해서 파일 읽기

■ 데이터 읽기

```
> fruits2 <- read.csv.sql("data-files/fruits-sql.csv", sql = "SELECT *  
FROM file WHERE Year = 2008")
```

Loading required package: tcltk

경고메시지(들):

사용되지 않는 커넥션 3 (data-files\fruits-sql.csv)를 닫습니다

```
> fruits2
```

	Fruit	Year	Location	Sales	Expenses	Profit	Date
1	Apples	2008	West	98	78	20	2008-12-31
2	Oranges	2008	East	96	81	15	2008-12-31
3	Bananas	2008	East	85	76	9	2008-12-31

Excel 파일 데이터 읽기

■ 패키지 설치

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

Excel 파일 데이터 읽기

▪ xls 파일 데이터 읽기

```
> df_exam <- read_excel("data-files/excel_exam.xlsx")
```

```
> df_exam <- as.data.frame(df_exam)
```

```
> df_exam
```

	id	class	math	english	science
1	1	1	50	98	50
2	2	1	60	97	60
3	3	1	45	86	78
4	4	1	30	98	58
5	5	2	25	80	65
6	6	2	50	89	98
7	7	2	80	90	45
8	8	2	90	78	25

```
... (생략)
```

Rdata 파일에 데이터 쓰고 읽기

- Rdata 파일

- R 전용 데이터 파일 (.rdata, .rda)
- R에서 읽고 쓰는 속도가 빠르고 용량이 작다

- RData 파일에 데이터 저장 → save() 함수 사용

```
> save(df_exam, file = "data-files/df_exam.rda")
```

Rdata 파일에 데이터 쓰고 읽기

- RData 파일에서 데이터 읽기 → `load()` 함수 사용

```
> rm(df_exam)
> df_exam
Error: object 'df_exam' not found
> load("data-files/df_exam.rda")
> df_exam
```

	id	class	math	english	science
1	1	1	50	98	50
2	2	1	60	97	60
3	3	1	45	86	78
4	4	1	30	98	58
5	5	2	25	80	65
...	(생략)				

클립보드의 내용을 사용해서 데이터 프레임 생성

- 파일의 내용을 Ctrl + C로 복사하고 아래 구문 실행

```
> fruits6 <- read.delim("clipboard", header = T)
> fruits6
```

	no	name	price	qty
1	1	apple	500	6
2	2	banana	400	4
3	3	peach	300	3
4	4	berry	200	2

(참고) Mac은 `read.delim(pipe("pbpaste"), header = T)`

데이터베이스 연동

- Oracle, MySQL 등 다양한 데이터 베이스 연동 지원
 - 관련 패키지 : RODBC, RJDBC 등 사용 가능
- 패키지 설치
 - RJDBC

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

데이터베이스 연동

■ MySQL 데이터베이스 연동

- working-directory에 mysql jdbc driver 복사

```
driver2 <- JDBC("com.mysql.cj.jdbc.Driver",
               "jdbc-drivers/mysql-connector-java-8.0.13.jar",
               identifier.quote = "'")
conn2 <- dbConnect(driver2,
                  "jdbc:mysql://127.0.0.1:3306/employees?serverTimezone=UTC",
                  "instructor", "Pa$$w0rd")
rs2 <- dbGetQuery(conn2, "select * from employees limit 1, 10")
rs2
```

	emp_no	birth_date	first_name	last_name	gender	hire_date
1	10002	1964-06-02	Bezalel	Simmel	F	1985-11-21
2	10003	1959-12-03	Parto	Bamford	M	1986-08-28
3	10004	1954-05-01	Chirstian	Koblick	M	1986-12-01

..... 생략