*범용 AI-AutoReporting 개발 매뉴얼

목 차

2019.6.20

1. 언어 및 실행 환경 설치

▣ 프래그램 언어

- R
- 오픈소스
- 다운로드 사이트 : https://cran.r-project.org/bin/windows/base/

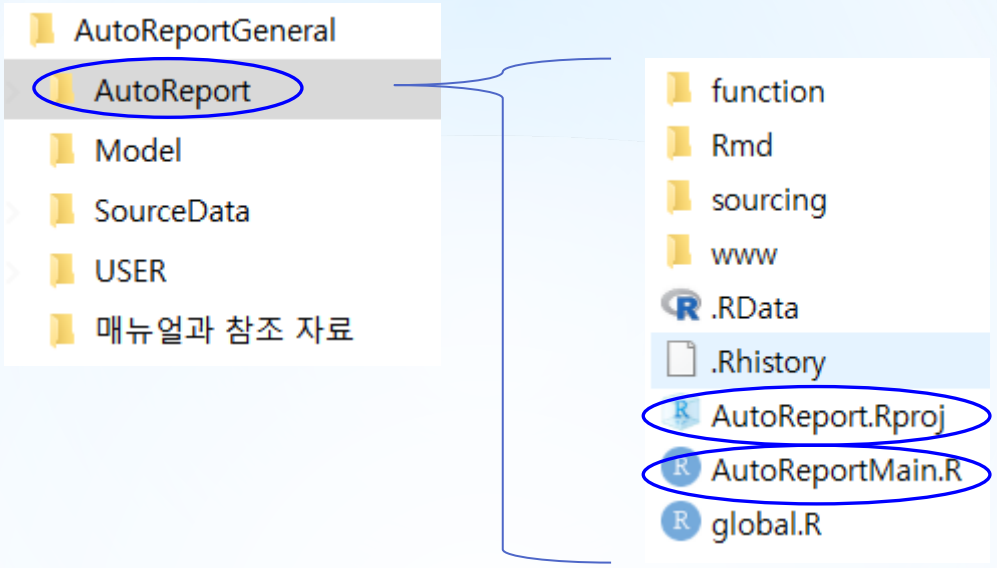
▣ 통합 개발 및 실행 환경

- RStudio
- 오픈소스
- 다운로드 사이트 : https://www.rstudio.com/products/rstudio/download/

▣ 배포 USB의 “download” 폴더에 윈도우용 상기 2개의 설치 파일 들어 있음

2. 자동화 프로그램 폴더 구조

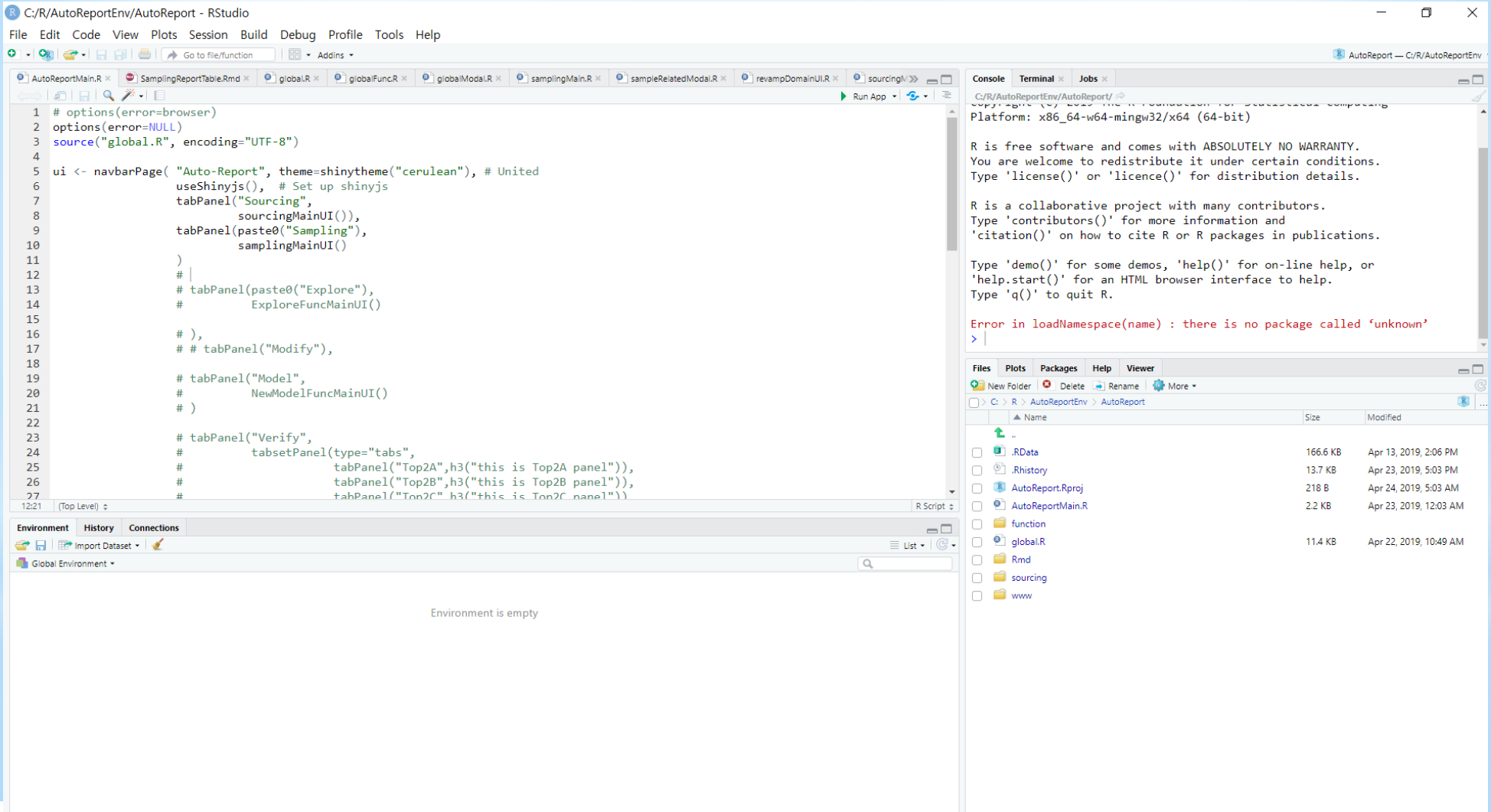
▣ 폴더 구조



- AutoReportGeneral : 최상위 폴더로 다섯 개의 하위 폴더 보유
  - AutoReport : 자동화 프로그램 코드 등을 저장하는 폴더
    - AutoReport.Rproj 파일 : Rstudio 개발 및 실행 환경 구축용 파일
    - AutoReportMain.R 파일: 자동화 프로그램의 메인 파일
  - Model: 예측 모델이 저장되는 폴더
  - SourceData: 맞춤용 데이터 파일이 저장되는 폴더
  - USER: 범용 데이터 파일이나, 리포트 파일이 저장되는 폴더
  - 매뉴얼과 참조자료 : 매뉴얼과 참조자료가 저장되는 폴더

3. 자동화 프로그램 활용 환경(Rstudio) 실행

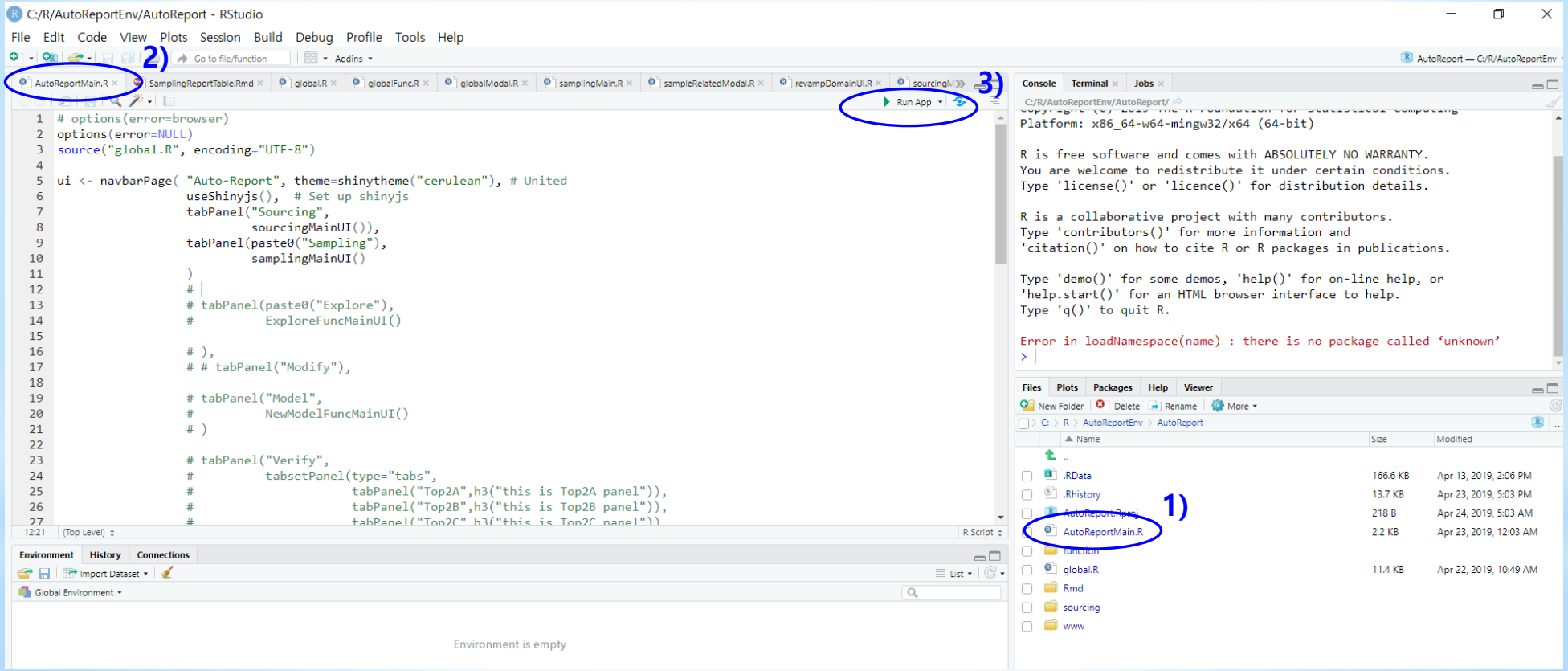
▣ 활용 환경 실행 ● AutoReport.Rproj 파일을 더블클릭 → 활용 환경 RStudio



4. 자동화 프로그램 실행

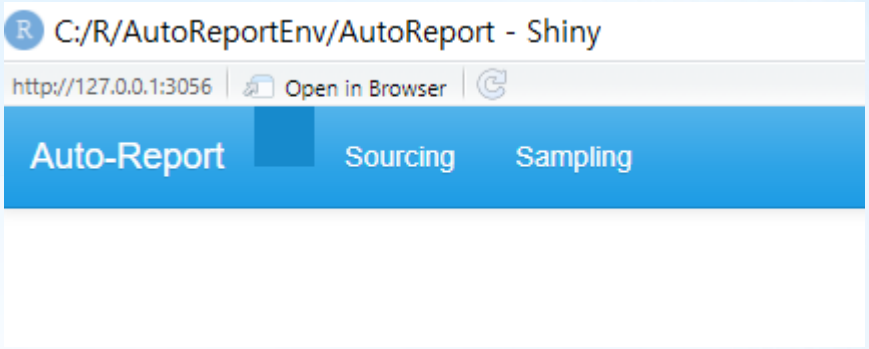
1) 4사분면의 AutoReportMain.R을 클릭

2) 2사분면의 코드 입력창에 AutoReportMain.R 파일이 보임



3) 코드 입력창의 Run App 버튼을 클릭하면 자동화 프로그램 실행 : 뒷페이지 프로그램 화면 참조

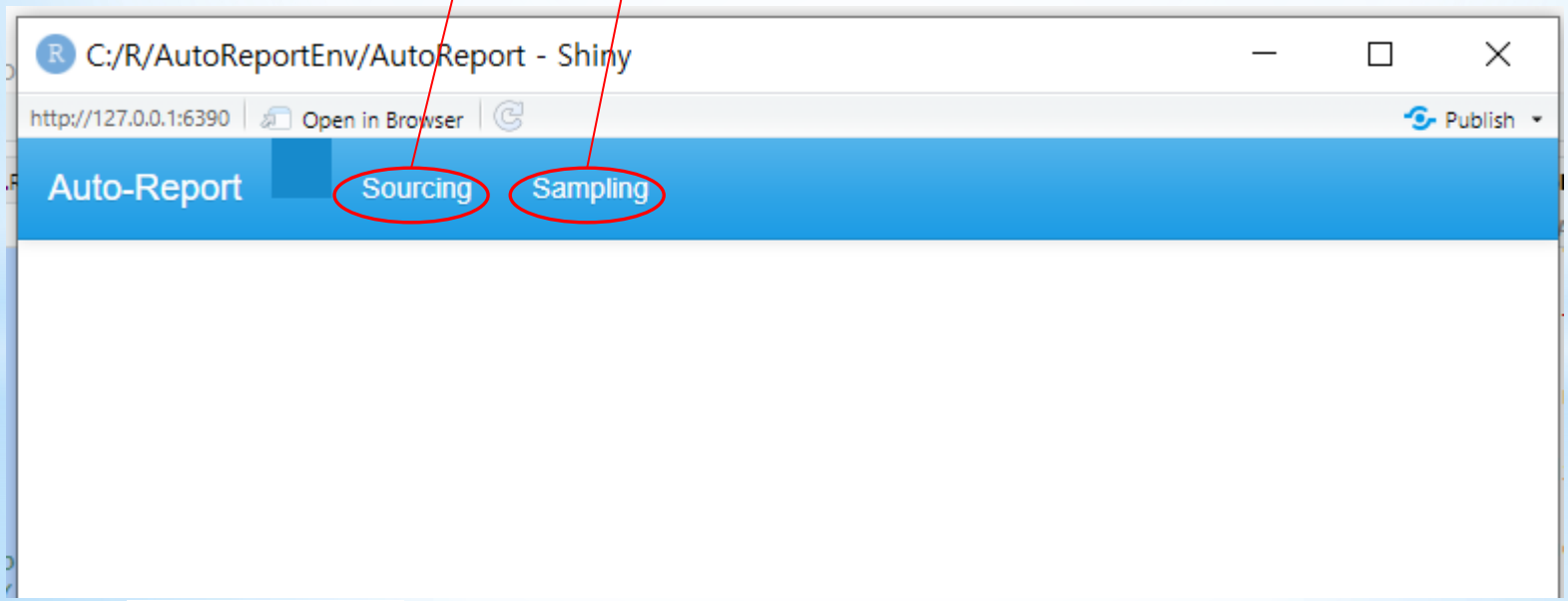
4) 실행된 자동화 프로그램 화면



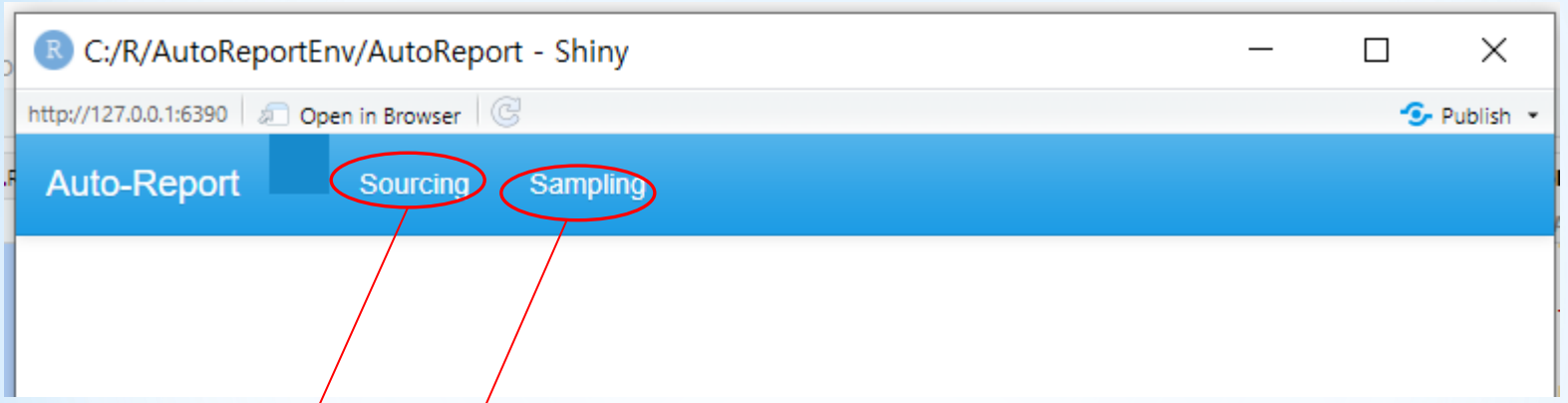
5-1. 메인 메뉴용 Code

▣ 메인 메뉴 생성 Code

```
ui <- navbarPage( "Auto-Report", theme=shinytheme("cerulean"), # United
                  useShinyjs(),  # Set up shinyjs
                  tabPanel("Sourcing",
                           sourcingMainUI()),
                  tabPanel(paste0("Sampling"),
                           samplingMainUI()
                  )
)
```



▣ 메인 메뉴 선택시 실행 Code




```
server <- function(input, output, session) {

  output$sum1 <- renderPrint({
    summary(cars)
  })

  sourcingMain(input, output, session)
  samplingMain(input, output, session)
}
```

5-2. 서브 메뉴용 Code

▣ Sourcing용 서브메뉴 생성 Code

```
sourcingMainUI <- function() {
  tabsetPanel( type="tabs",
    tabPanel("SourcingSub1",
      fluidPage(
        fluidRow(
          column(2, actionButton("renderReportSourcing", "리포트") ),
          includeCSS("www/html/globalModal.css"),
          column(2, actionButton("renderReportCommonSource", "범용 리포트") ),
          column(6, tags$h4("::::::::") ),
          column(2, actionButton("treatVar", "변수 전처리") )
        ),
        tags$hr(),
        fluidRow(
          column(3,
              switch(sourcingCat,
                  RExample = {radioButtons("source", "원천 데이타",
                      c("empty","수입세 환급"="importTaxRefund","mtcars", "diamonds","범용"="EXCEL", "clipboard" ))}
          ),
          column(9, erbatimTextOutput("strDFsource"))
        )
      )
    )
  )
}
```

5-2. 서브 메뉴용 Code

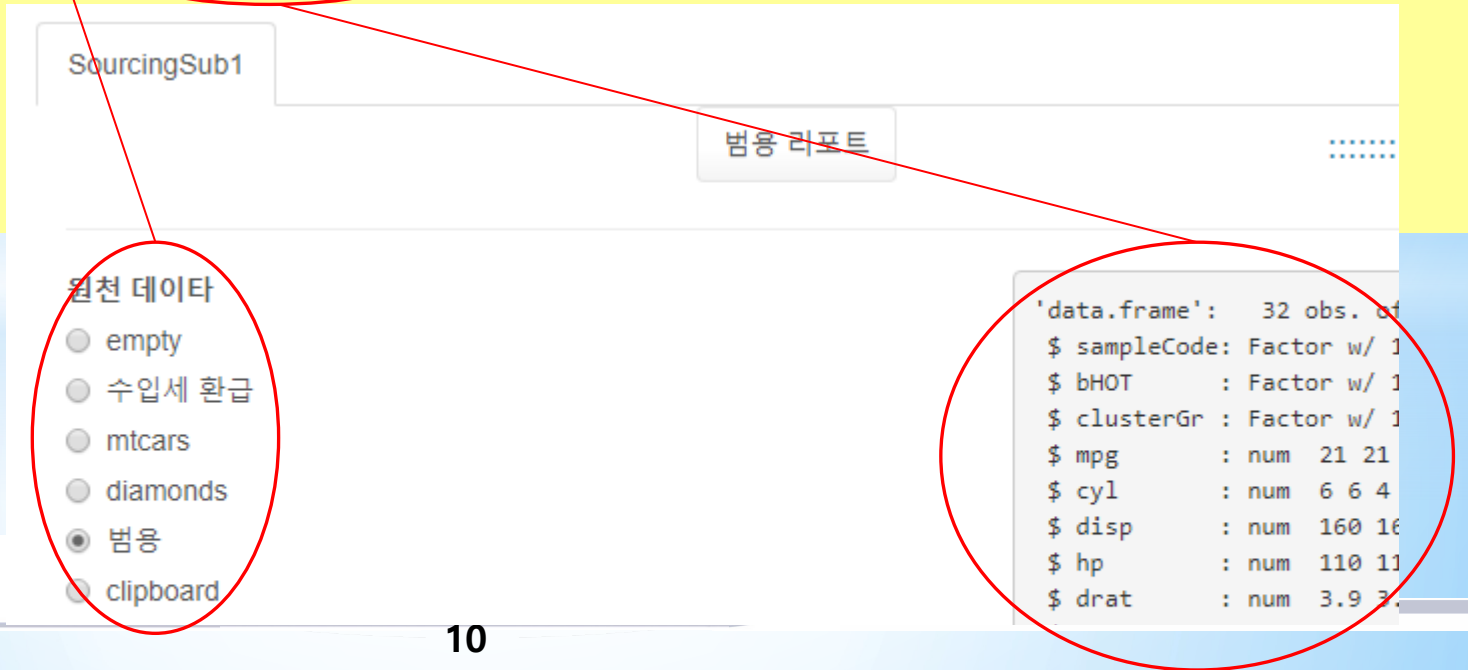
▣ Sourcing용 서브메뉴 생성 Code

```
sourcingMainUI <- function() {
  tabsetPanel( type="tabs",
    tabPanel("SourcingSub1",
      fluidPage(
        fluidRow(
          column(2, actionButton("renderReportSourcing", "리포트") ),
          includeCSS("www/html/globalModal.css"),
          column(2, actionButton("renderReportCommonSource", "범용 리포트") ),
          column(6, tags$h4(":::::::") ),
          column(2, actionButton("treatVar", "변수 전처리") )
       ),
    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
}
```

5-2. 서브 메뉴용 Code

▣ Sourcing용 서브메뉴 생성 Code

```
sourcingMainUI <- function() {
  tabsetPanel( type="tabs",
    tabPanel("SourcingSub1",
      fluidPage(
        ~~~~~~~~~~~~~~~~~
        fluidRow(
          column(3,
              switch(sourcingCat,
                  RExample = {radioButtons("source", "원천 데이타",
                      c("empty","수입세 환급"="importTaxRefund","mtcars", "diamonds","범용"="EXCEL", "clipboard" ))}
          ),
          column(9, verbatimTextOutput("strDFsource"))
        )
      )
    )
  )
}
```



5-3. 원천데이터 Sourcing 대화창용 Code

▣ 원천데이터 라디오버튼 클릭 대응 Code

```
output$strDFsource <- renderPrint({
    chosenDFSourceFile <<- ""
    show("renderReportSourcing")
    hide("treatVar")

      switch(input$source,
             importTaxRefund = {
               source("sourcing/importTaxRefund/loadSource.R", encoding="UTF-8")
               loadSource()
             },
             mtcars = {
               source("sourcing/mtcars/loadSource.R", encoding="UTF-8")
               loadSource()
             },
             diamonds = {
               source("sourcing/diamonds/loadSource.R", encoding="UTF-8")
               loadSource()
             },
             EXCEL = {
               show("treatVar")
               source("sourcing/EXCEL/loadSource.R", encoding="UTF-8")
               loadSource()

             },
    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

5-3. 원천데이터 Sourcing 대화창용 Code

▣ 원천데이터 라디오버튼 클릭 대응 Code

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

                clipboard = {
                DFSource <<- read.delim("clipboard", stringsAsFactors = T)
                # 빈 InsertReqInputUI 함수 정의
                source("sourcing/clipboard/function/InsertReqInputUI.R", encoding="UTF-8")
            },
            empty = {
                return()
            },
            {}

        )
        bHOT <- rep("Normal", NROW(DFSource))
        clusterGr <- rep("1", NROW(DFSource))
        sampleCode <- rep("remains", NROW(DFSource))
        DFSource <<- cbind(sampleCode, bHOT, clusterGr,  DFSource )
        reactDFSource(DFSource)
        curSampleExplore <<- DFSource
        str(DFSource)

    })
```

5-3. 원천데이터 Sourcing 대화창용 Code

▣ 범용데이터 선정 대화창 생성 Code

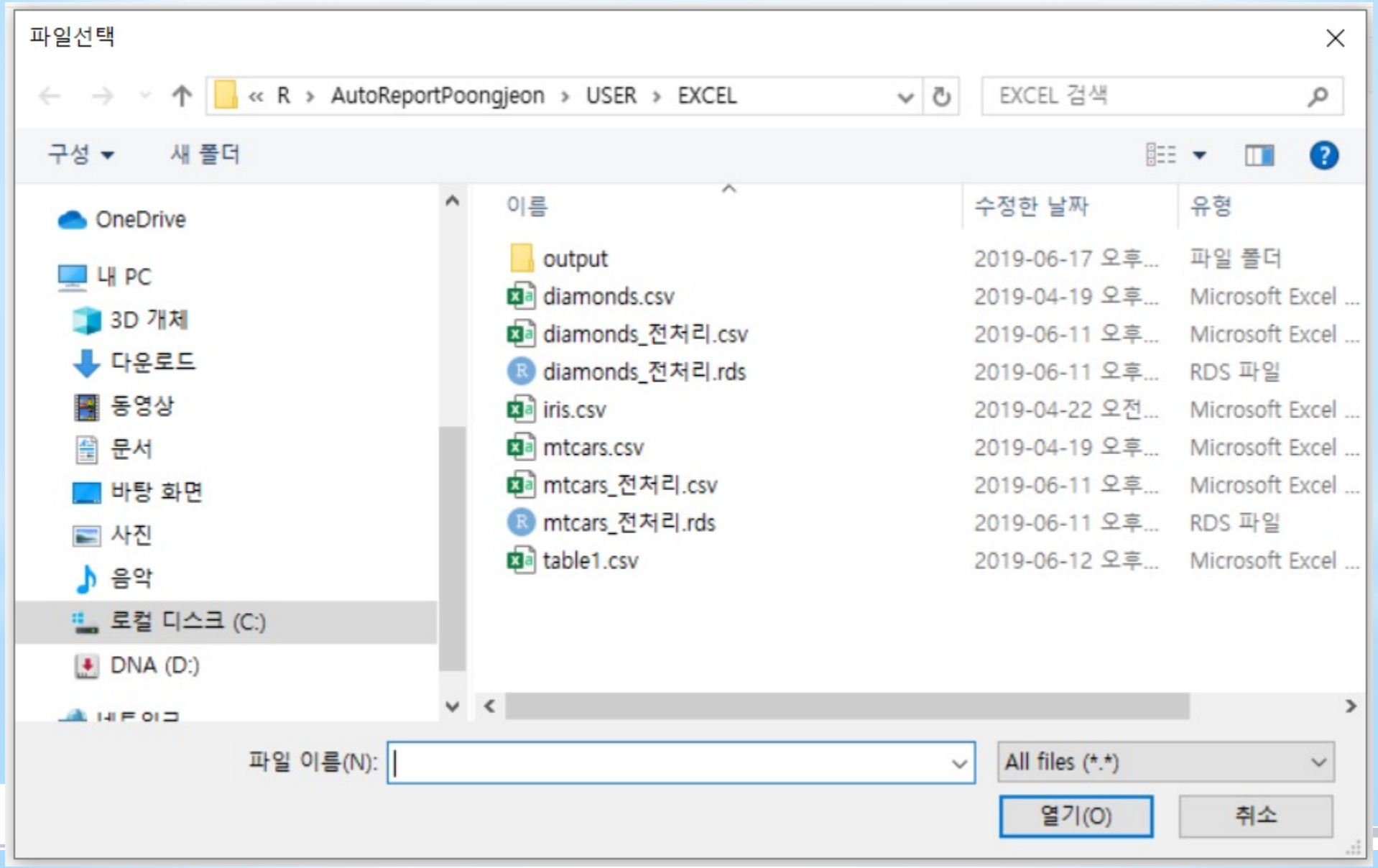
```
loadSource <- function() {
  projectWD <- getwd()
  setwd("..")
  dirPath <- paste0(getwd(),"/USER/EXCEL")
  setwd(dirPath)
  # Tell R to sleep until the current directory matches the expected directory
  while(getwd() != normalizePath(dirPath, winslash="/")) {
    Sys.sleep(0.02)
  }
  filePath <- file.choose()

  chosenDFSourceFileExt <<- str_split(filePath, "₩₩₩₩")[[1]][length( str_split(filePath, "₩₩₩₩")[[1]])]
  chosenDFSourceFile <<- str_split(chosenDFSourceFileExt, "₩₩.")[[1]][1]
  chosenDFSourceExt <- str_split(chosenDFSourceFileExt, "₩₩.")[[1]][2]

  setwd("../../AutoReport")
  switch(chosenDFSourceExt,
         csv = {
           DFSource <<- read_csv(filePath)
         },
         rds = {
           DFSource <<- read_rds(filePath)
         }
~~~~~~~~~~~~~~~~~~~~~~~
```

5-3. 원천데이터 Sourcing 대화창용 Code

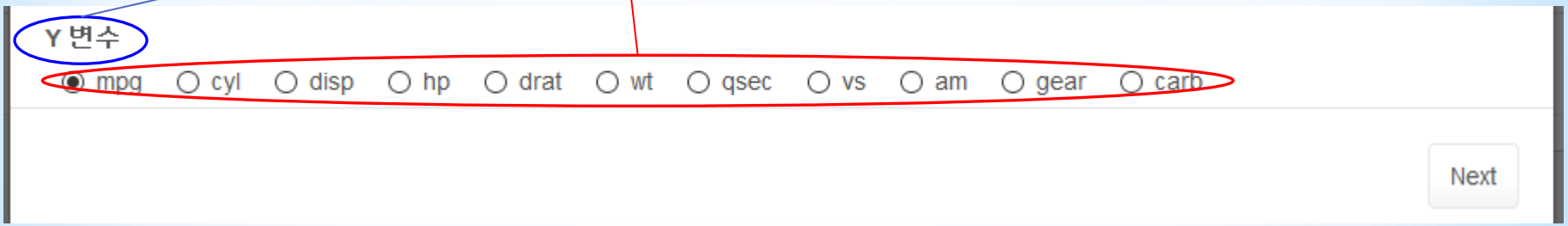
▣ 범용데이터 선정 대화창 생성 Code



(첨부 3) Auto-Reporting 주요 Code

▣ Radio Button 대화창 생성 Code

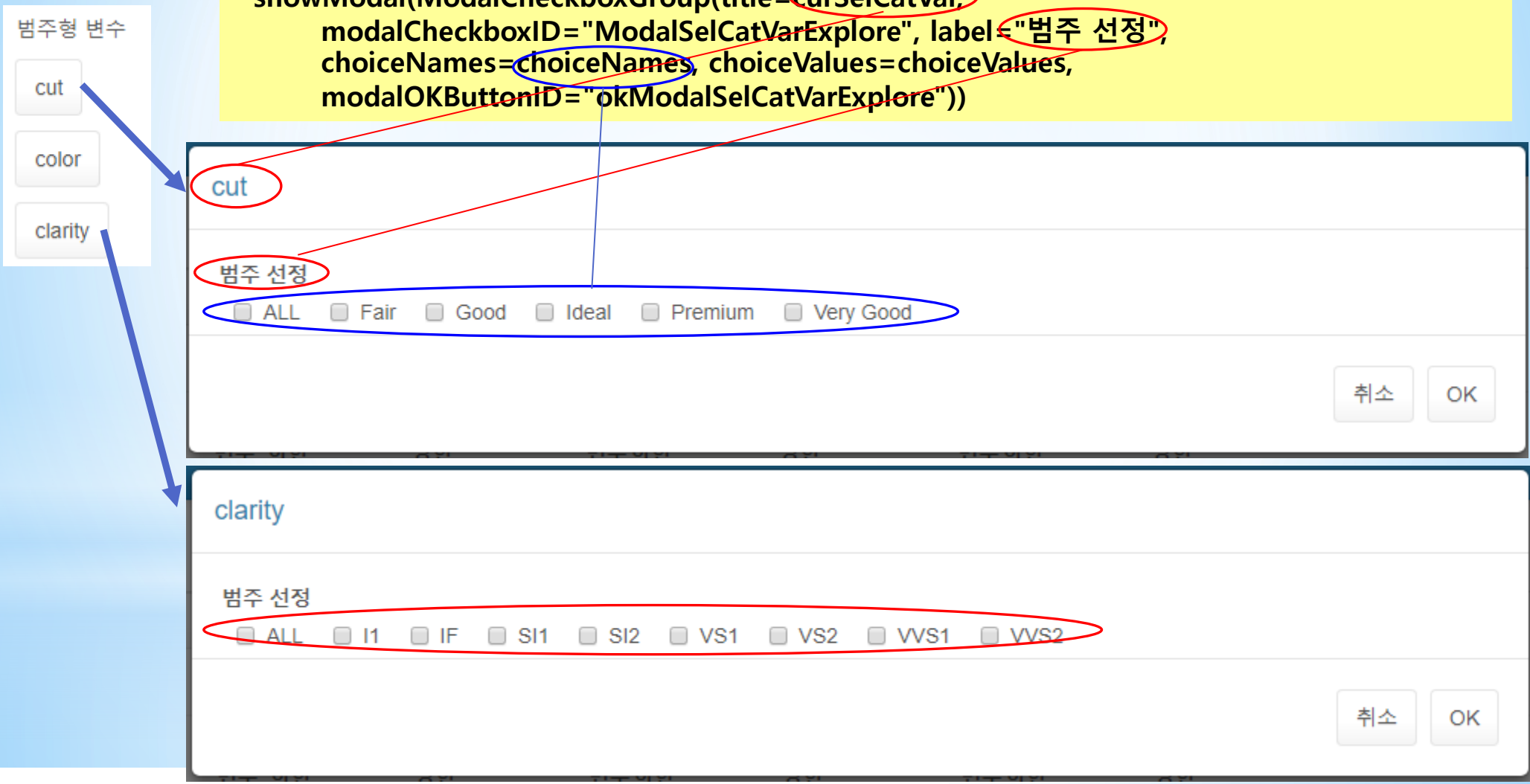
```
showModal(ModalRadioButtons(numVar, numVar, "okMCP","Y 변수", failed = FALSE))
```



(첨부 3) Auto-Reporting 주요 Code

▣ Checkbox 대화창 생성 Code

```
showModal(ModalCheckboxGroup(title=curSelCatVar,
        modalCheckboxID="ModalSelCatVarExplore", label="범주 선정",
        choiceNames=choiceNames, choiceValues=choiceValues,
        modalOKButtonID="okModalSelCatVarExplore"))
```



(첨부 3) Auto-Reporting 주요 Code

▣ 리포트 생성 R Code

```
params <- list(df=dfReportCommon)
outputFileName <- paste0("commonDescriptiveReport", fromReportCommon, chosenDFSourceFile, ".html")
rmarkdown::render("Rmd/commonDescriptiveReport.Rmd", output_file = outputFileName,
                  output_dir = pathHTMLReport,
                  params = params,
                  envir = new.env(parent = globalenv()))
```

리포트를 생성하는 Rmarkdown Code

생성된 리포트 파일명
(예) commonDescriptiveReport_source_mtcars.html
리포트 종류_분석 단계_데이타파일명

(첨부 3) Auto-Reporting 주요 Code

▣ 리포트 생성 Rmarkdown Code

````
Scatter1 {data-navmenu="Scatter Plot"}
=====================================

Row {data-height=500}
-------------------------------------

###
```{r scatter_1}
renderScatterPlot <- function(x, y, color=NULL, size=NULL, shape=NULL) {
  ~~~~~~~~~~~~~
}
if(!is.na(x=aesList[["x"]][1]))
  renderScatterPlot(x=aesList[["x"]][1], y=aesList[["y"]], color=aesList[["color"]],
                    size=aesList[["size"]], shape=aesList[["shape"]])
```

###
```{r scatter_2}
if(!is.na(x=aesList[["x"]][2]))
  renderScatterPlot(x=aesList[["x"]][2], y=aesList[["y"]], color=aesList[["color"]],
                    size=aesList[["size"]], shape=aesList[["shape"]])
```
````
  ~~~~~~~~~~~~~

(첨부 3) Auto-Reporting 주요 Code

▣ 리포트 생성 Rmarkdown Code

````
Row {data-height=500}
-----------------------------------------

###
```{r, scatter_3}
if(!is.na(x=aesList[["x"]][3]))
  renderScatterPlot(x=aesList[["x"]][3], y=aesList[["y"]], color=aesList[["color"]],
                    size=aesList[["size"]], shape=aesList[["shape"]])
```


###
```{r, scatter_4}
if(!is.na(x=aesList[["x"]][4]))
  renderScatterPlot(x=aesList[["x"]][4], y=aesList[["y"]], color=aesList[["color"]],
                    size=aesList[["size"]], shape=aesList[["shape"]])
```
````

(첨부 3) Auto-Reporting 주요 Code

▣ 리포트 페이지