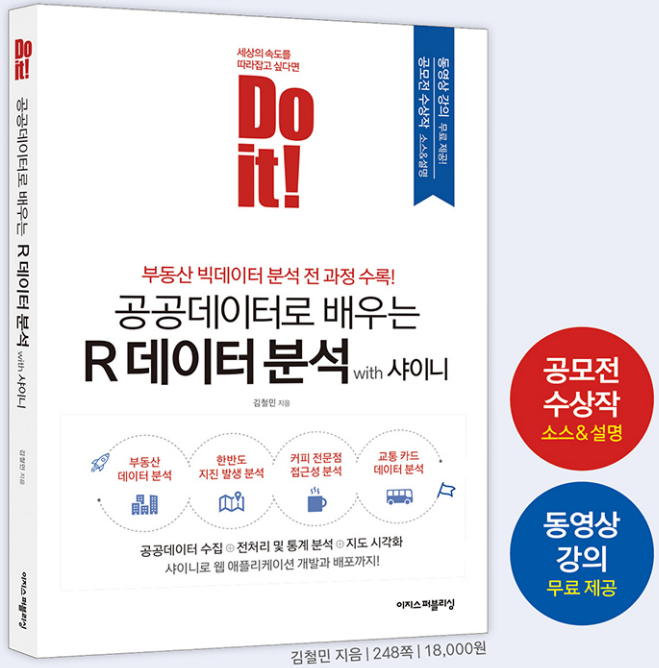
R로 공공데이터를 분석하는 전 과정 실습!

공모전 수상작으로 배우는 **R 데이터 분석**



공모전 수상작 소스&설명

동영상 강의 무료 제공

김철민 지음 | 248쪽 | 18,000원

샤이니는 데이터 분석 결과를 애플리케이션으로 만드는 개발 도구입니다. 이번 장에서는 샤이니의 기본 개념과 용어를 알아보고 작동 원리인 입출력, 반응성 그리고 레이아웃을 소개합니다.

09

샤이니 입문하기

09-1 처음 만나는 샤이니

1단계 샤이니 기본 구조 이해하기

Do it! 샤이니 기본 구조 이해 — 09_샤이니 입문.R

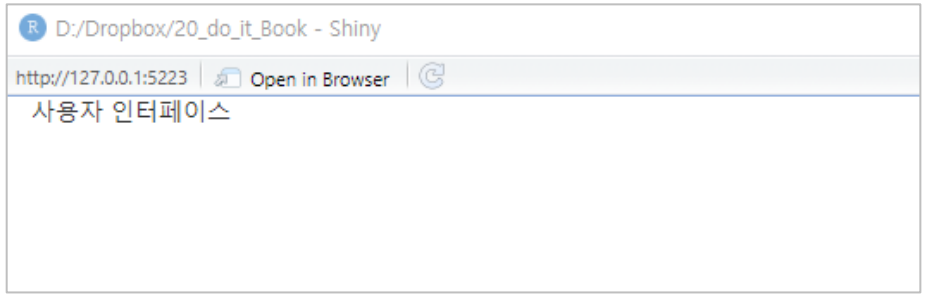
```
09: library(shiny)   # install.packages("shiny")
10: ui <- fluidPage("사용자 인터페이스")
11: server <- function(input, output, session){ }
12: shinyApp(ui, server)
```

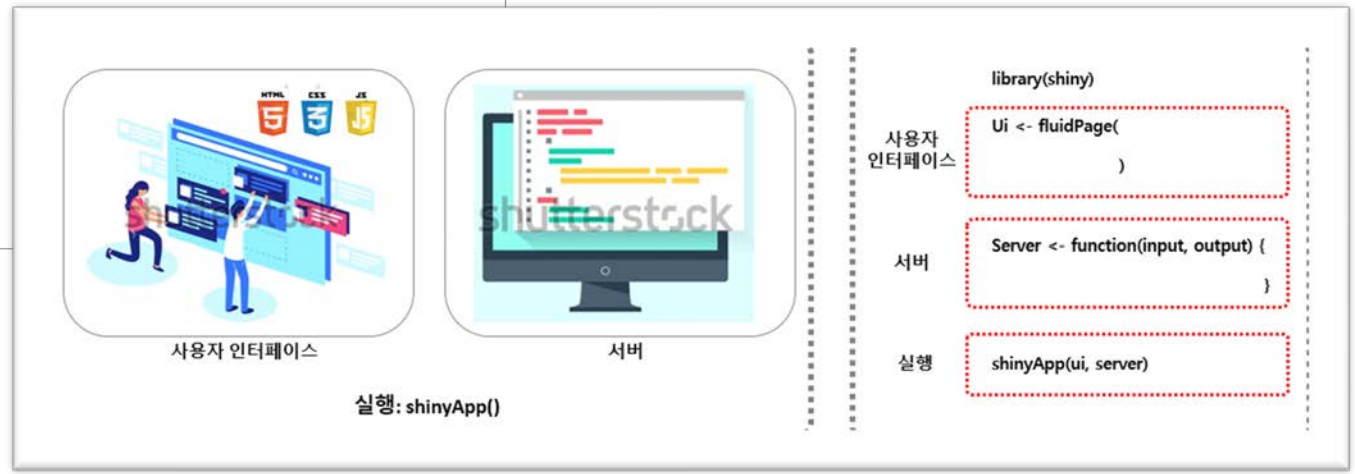
사용자 인터페이스

서버

실행

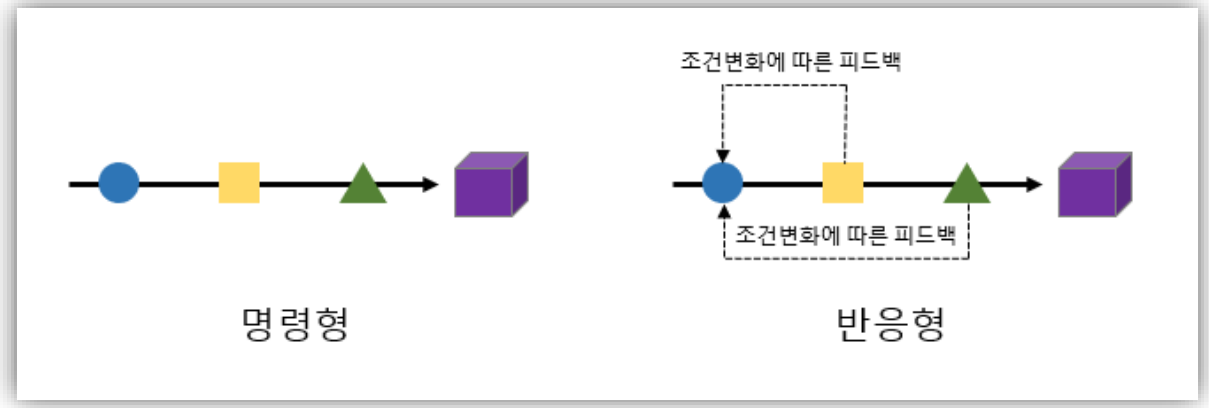
☞ 실행 결과





09-1 처음 만나는 샤이니

2단계 샘플 실행해 보기



Do it! 샤이니가 제공하는 샘플 확인하기 09_샤이니 입문.R

```
16: library(shiny)    # 라이브러리 등록
17: runExample()      # 샘플 보여주기
```

☞ 실행 결과

```
Valid examples are "01_hello", "02_text", "03_reactivity", (...생략...)
```

Do it! 첫 번째 샘플 실행하기 09_샤이니 입문.R

```
18: runExample("01_hello")    # 1번 샘플 실행
```

☞ 실행 결과

09-1 처음 만나는 샤이니

3단계 사용자 인터페이스 부분

Do it! 01_hello 샘플의 사용자 인터페이스 부분 — 09_샤이니 입문.R

```
27: library(shiny)       # 라이브러리 등록
28: ui <- fluidPage(     # 사용자 인터페이스 시작: fluidPage 정의
29:   titlePanel("샤이니 1번 샘플"),  # 제목 입력
30:   #---# 레이아웃 구성: 사이드바 패널 + 메인 패널
31:   sidebarLayout(
32:     sidebarPanel(  # 사이드바 패널 시작
33:       #--- 입력값: input$bins 저장
34:       sliderInput(inputId = "bins",           # 입력 아이디
35:                   label = "막대(bin) 개수:",   # 텍스트 라벨
36:                   min = 1, max = 50,          # 선택 범위(1-50)
37:                   value = 30)),               # 기본값 30
38:     mainPanel(   # 메인 패널 시작
39:       #---# 출력값: output$distPlot 저장
40:       plotOutput(outputId = "distPlot"))  # 차트 출력
41:   ))
```

타이틀 패널

사이도바 패널

메인 패널

09-1 처음 만나는 샤이니

4단계 서버 부분

Do it! 01_hello 샘플의 서버 부분 — 09_샤이니 입문.R

```
45: server <- function(input, output, session){
46:   #---# 랜더링한 플롯을 output 인자의 distPlot에 저장
47:   output$distPlot <- renderPlot({
48:     x <- faithful$waiting   # 분출 대기 시간 정보 저장
49:     #---# input$bins을 플롯으로 렌더링
50:     bins <- seq(min(x), max(x), length.out = input$bins + 1)
51:     #---# 히스토그램 그리기
52:     hist(x, breaks = bins, col = "#75AADB", border = "white",
53:          xlab = "다음 분출 때까지 대기 시간(분)",
54:          main = "대기 시간 히스토그램")
55:   })
56: }
57: #---# 실행
58: shinyApp(ui, server)
59: rm(list = ls())  # 메모리 정리하기
```

분출 데이터 X 변수로 저장

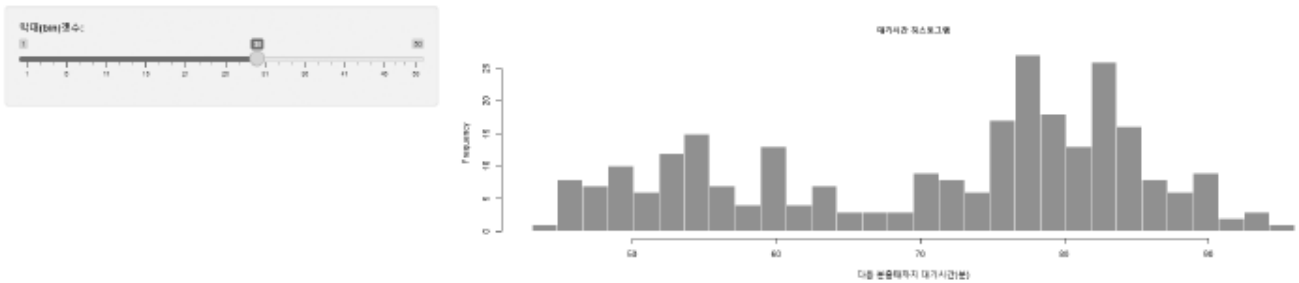
히스토그램 구간설정(bin)

히스토그램 그리기

실행

☞ 실행 결과

샤이니 1번 샘플



09-2 입력과 출력하기

1단계 입력받기 input$~

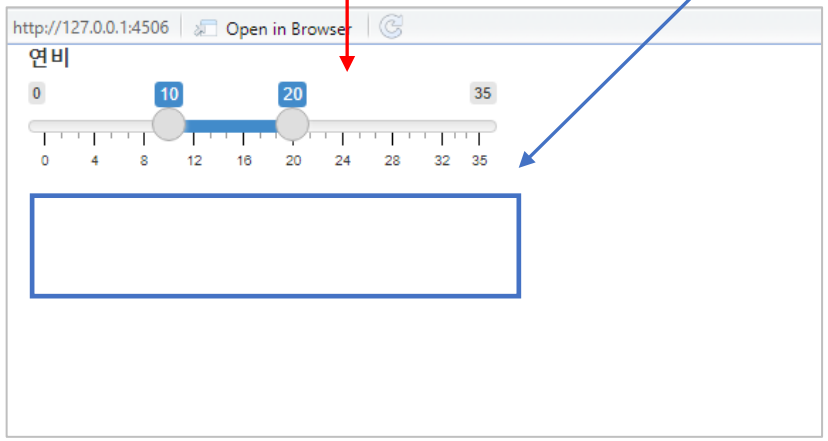
Do it! 데이터 입력 — 09_샤이니 입문.R

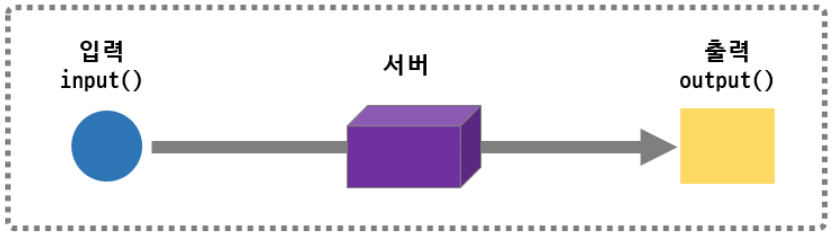
```
68: library(shiny)
69: ui <- fluidPage(
70:   sliderInput("range", "연비", min = 0, max = 35, value = c(0, 10)))   # 데이터 입력
71:
72: server <- function(input, output, session) {}    # 반응 없음
73:
74: shinyApp(ui, server)    # 실행
```

정의하지 않아서 반응이 없음

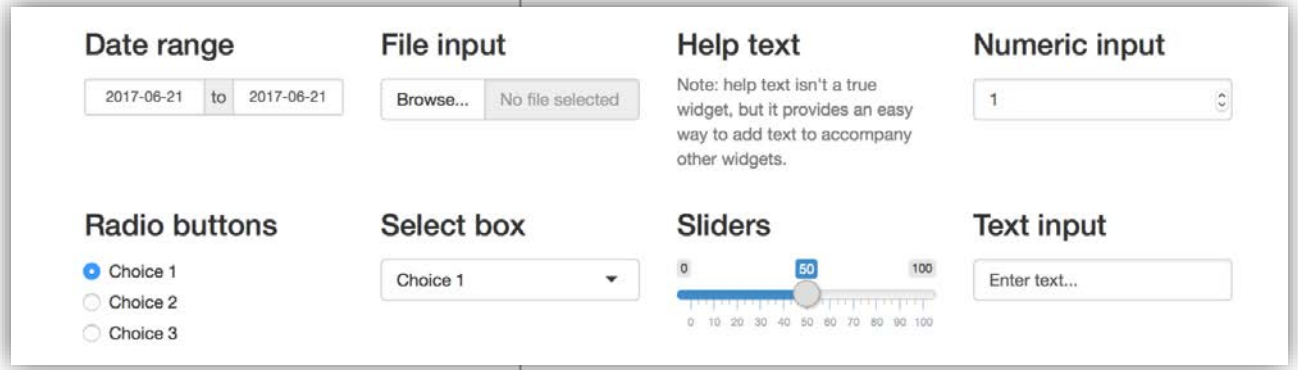
입력

☞ 실행 결과





다양한 입력모듈



09-2 입력과 출력하기

2단계 출력하기 output$~

Do it! 데이터 출력 — 09_샤이니 입문.R

```
78: library(shiny)
79: ui <- fluidPage(
80:   sliderInput("range", "연비", min = 0, max = 35, value = c(0, 10)),
81:   textOutput("value"))
82:
83: server <- function(input, output, session) {
84:     output$value <- renderText((input$range[1] + input$range[2]))}
85:
86: shinyApp(ui, server)
```

데이터 입력

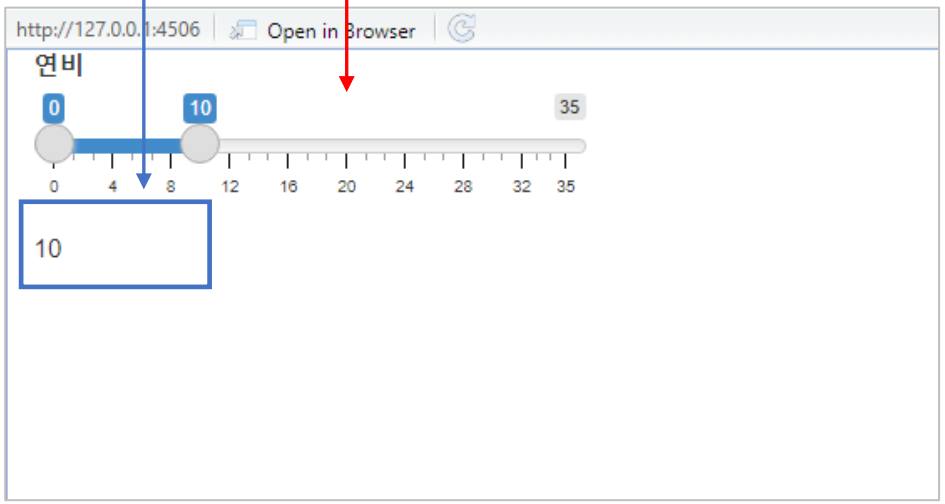
결괏값 출력

입력값 계산

출력

입력

☞ 실행 결과



09-2 입력과 출력하기

3단계 렌더링 함수의 중요성 render~()

Do it! 렌더링 함수의 중요성 09_샤이니 입문.R

```
90: library(shiny)
91: ui <- fluidPage(
92:  sliderInput("range", "연비", min = 0, max = 35, value = c(0, 10)),
93:  textOutput("value"))   # 출력
94:
95: server <- function(input, output, session) {
96:  output$value <- (input$range[1] + input$range[2])}
97:
98: shinyApp(ui, server)
```

데이터 입력

결괏값 갱신 안 됨

렌더링 함수가 없어서 오류 발생

renderText 없음

에러

☞ 실행 결과

```
Listening on http://127.0.0.1:3738
Warning: Error in : Can't access reactive value 'range'….
i Do you need to wrap inside reactive() or observe()?
 55: <Anonymous>
Error : Can't access reactive value 'range' outside of reactive consumer.
i Do you need to wrap inside reactive() or observe()?
```

1단계 데이터 준비하기

Do it! 데이터 준비 09_샤이니 입문.R

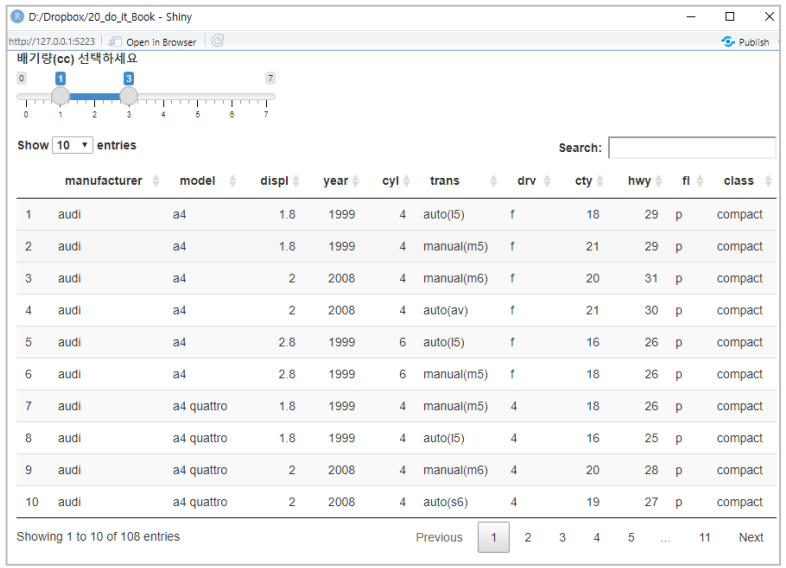
```
107: library(DT)        # install.packages("DT")
108: library(ggplot2)   # install.packages("ggplot2")
109: mpg <- mpg
110: head(mpg)
```

☞ 실행 결과

```
# A tibble: 6 x 11
  manufacturer model displ  year   cyl trans   drv     cty   hwy fl    class
  <chr>        <chr> <dbl> <int> <int> <chr>   <chr> <int> <int> <chr> <chr>
1 audi         a4      1.8  1999     4 auto(l~ f        18    29 p     comp~
2 audi         a4      1.8  1999     4 manual~ f        21    29 p     comp~
```

09-3 반응형 웹 애플리케이션 만들기

2단계 반응식 작성하기



Do it! 반응식 작성 — 09_샤이니 입문.R

```
114: library(shiny)
115: ui <- fluidPage(
116:   sliderInput("range", "연비", min = 0, max = 35, value = c(0, 10)), # 데이터 입력
117:   DT::dataTableOutput("table"))    # 출력
118:
119: server <- function(input, output, session){
120:   #---# 반응식
121:   cty_sel = reactive({
122:     cty_sel = subset(mpg, cty >= input$range[1] & cty <= input$range[2])
123:     return(cty_sel)})
124:   #---# 반응 결과 렌더링
125:   output$table <- DT::renderDataTable(cty_sel()) }
126:
127: shinyApp(ui, server)
```

렌더링 결과

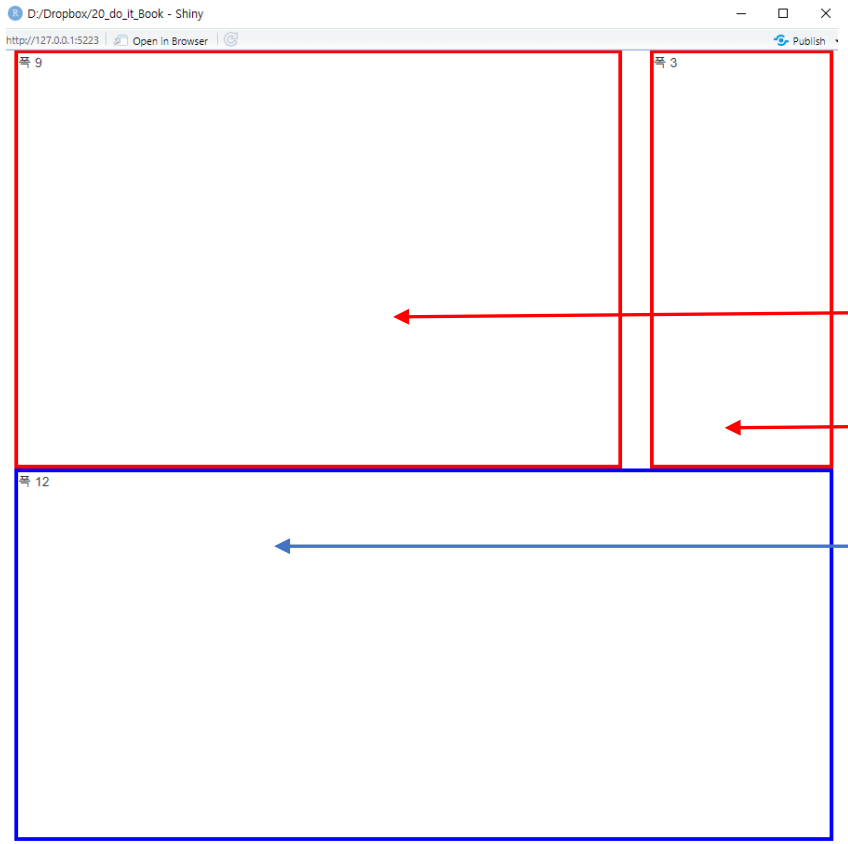
입력값 전달

반응식 결과 전달

에러

09-4 레이아웃 정의하기

1단계 단일 페이지 레이아웃

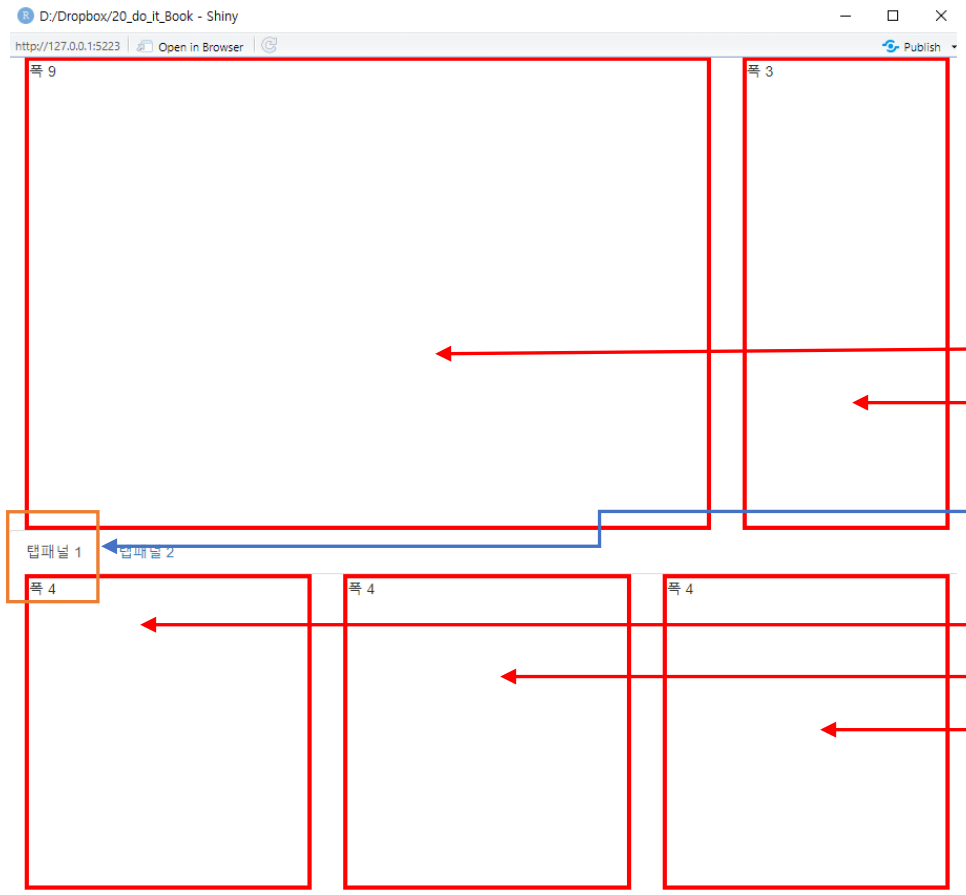


Do it! 단일 페이지 화면 — 09_샤이니 입문.R

```
136: library(shiny)
137: #---# 전체 페이지 정의
138: ui <- fluidPage(
139:  #---# 행 row 구성 정의
140:  fluidRow(
141:       #---# 첫 번째 열: 빨강(red) 박스로 높이 450 픽셀, 폭 9
142:       column(9, div(style = "height:450px;border: 4px solid red;","폭 9")),
143:       #---# 두 번째 열: 보라(purple) 박스로 높이 450 픽셀, 폭 3
144:       column(3, div(style = "height:450px;border: 4px solid purple;","폭 3")),
145:       #---# 세 번째 열: 파랑(blue) 박스로 높이 400 픽셀, 폭 12
146:       column(12, div(style = "height:400px;border: 4px solid blue;","폭 12"))))
147: server <- function(input, output, session) {}
148: shinyApp(ui, server)
```

09-4 레이아웃 정의하기

2단계 탭 페이지 추가하기



Do it! 탭 페이지 추가 — 09_샤이니 입문.R

```
152: library(shiny)
153: ui <- fluidPage(
154:  fluidRow(
155:   column(9, div(style = "height:450px;border: 4px solid red;","폭 9")),
156:   column(3, div(style = "height:450px;border: 4px solid red;","폭 3")),
157:  #---# 탭 패널 1~2번 추가
158:  tabsetPanel(
159:   tabPanel("탭1",
160:    column(4, div(style = "height:300px;border: 4px solid red;","폭 4")),
161:    column(4, div(style = "height:300px;border: 4px solid red;","폭 4")),
162:    column(4, div(style = "height:300px;border: 4px solid red;","폭 4")), ),
163:   tabPanel("탭2",div(style ="height:300px;border: 4px solid blue;","폭 12")))))
164: server <- function(input, output, session) {}
165: shinyApp(ui, server)
```