8.3 Come analizzare i dati: introduzione a Shiny

Insegnamento di Informatica

Elisabetta Ronchieri

Corso di Laurea di Economia, Universitá di Ferrara

I semestre, anno 2014-2015

Argomenti

Shiny

Basic Example

Examples with some complex operations

Others

Argomenti

Shiny

Basic Example

Examples with some complex operations

Others

What is Shiny

- Shiny is an open-sourced by RStudio 11/2012 on CRAN.
- It represents a new model for Web-accessible R code.
- It is able to generate basic web user interfaces.
- It uses "The new HTTP".
- It is built on a "Reactive Programming" model.
- It supports custom inputs and outputs.

What is Shiny

- With Shiny you can very easily:
  - create Web input;
  - create form that calls R and thus your R code
  - display the results.

Reactive programming 1

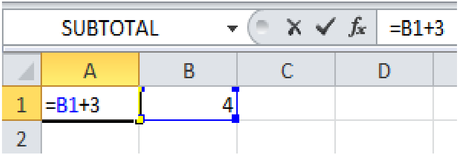
- Given the following settings to the *a* and *b* variables, determine the new value of *b*:

  ```
  a <- 3
  b <- a + 2
  a <- 7
  ```

- In imperative programming, $b = 5$.
- In reactive programming, $b = 9$.
- R is an imperative programming language. **It will always get** $b = 5$.

Reactive programming 2

- In the spreadsheet the use of the $= B1 + 4$ formula is an example of reactive programming.



SUBTOTAL   =B1+3

	A	B	C	D
1	=B1+3	4		
2				

- Changing the value in the field B1 triggers a new elaboration of the A1 value.

Start with Shiny

- Make sure you have the latest release of R installed
- If on windows, make sure that you have installed the **shiny** package.

  ```
  install.packages("shiny")
  libray(shiny)
  ```
- See tutorial at
  http://rstudio.github.io/shiny/tutorial/

Argomenti

Shiny

Basic Example

Examples with some complex operations

Others

Basic Example

- A shiny project is a directory containing at least two parts:
  1. One named UI.R (for user interface) that controls how it looks.
  2. One named Server.R that controls what it does.

`http://trestletechnology.net:3838/simpleGeyeser/`

`https://github.com/trestletech/shiny-sandbox/tree/master/simpleGeyeser`

UI.R

```
library(shiny)
shinyUI(pageWithSidebar(
 headerPanel("Hello Shiny!"),
 sidebarPanel(h3(’Sidebar text’)),mainPanel(h3(’Main Panel text’))
))
```

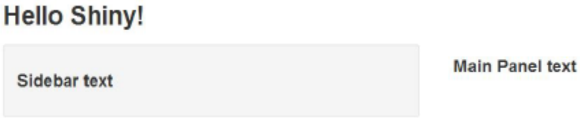
Sever.R

```
library(shiny)
shinyServer(
function(input, output) {
}
)
```

To run it

- Change to the directories with these files and type runApp()
- Put the path to the directory as an argument
- It should open a browser window with the app running.

Shiny output



Hello Shiny!

Sidebar text

Main Panel text

Argomenti

Shiny

Basic Example

Examples with some complex operations

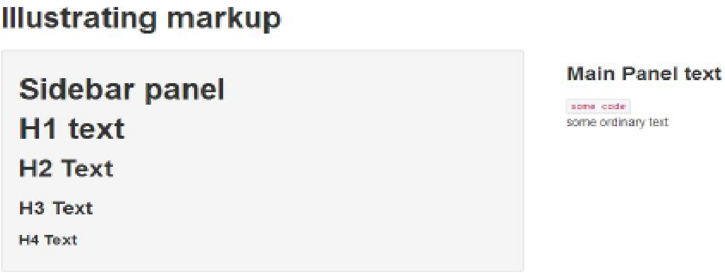
Others

UI.R

Illustrating markup:

```
shinyUI(pageWithSidebar(
headerPanel("Illustrating markup"),
sidebarPanel(
h1('Sidebar panel'),
h1('H1 text'),
h2('H2 Text'),
h3('H3 Text'),
h4('H4 Text')
),
mainPanel(
h3('Main Panel text'),
code('some code'),
p('some ordinary text')
)
))
```

Shiny output



Illustrating markup

Sidebar panel

H1 text

H2 Text

H3 Text

H4 Text

Main Panel text

`some code`

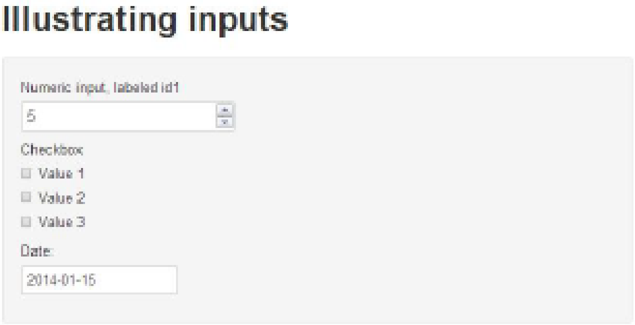
some ordinary text

UI.R

Illustrating inputs:

```
shinyUI(pageWithSidebar(
headerPanel("Illustrating inputs"),
sidebarPanel(
numericInput('id1', 'Numeric input, labeled id1', 0,
min = 0, max = 10, step = 1),
checkboxGroupInput("id2", "Checkbox",
c("Value 1" = "1",
"Value 2" = "2",
"Value 3" = "3")),
dateInput("date", "Date:")
),
mainPanel(
)
))
```

Shiny output



Illustrating inputs

Numeric input, labeled id1

5

Checkbox

□ Value 1

□ Value 2

□ Value 3

Date:

2014-01-15

UI.R

Illustrating outputs:

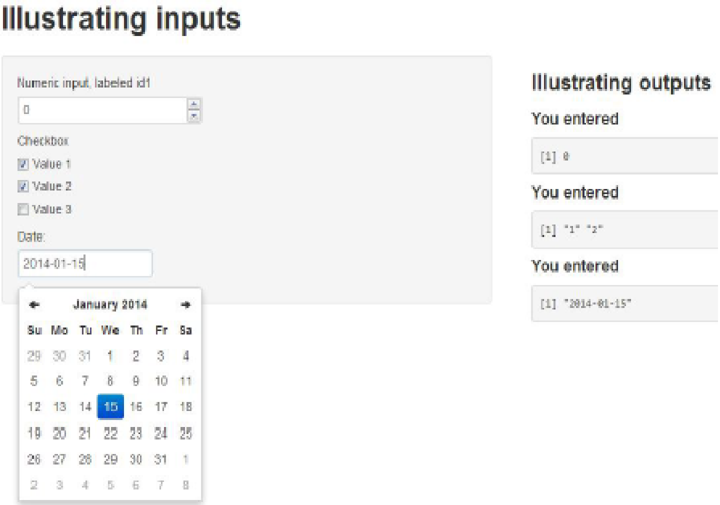
```
mainPanel(
h3('Illustrating outputs'),
h4('You entered'),
verbatimTextOutput("oid1"),
h4('You entered'),
verbatimTextOutput("oid2"),
h4('You entered'),
verbatimTextOutput("odate")
)
```

Server.R

Illustrating outputs:

```
shinyServer(
function(input, output) {
output$oid1 <- renderPrint({input$id1})
output$oid2 <- renderPrint({input$id2})
output$odate <- renderPrint({input$date})
}
)
```

Shiny output



Illustrating inputs

Numeric input, labeled id1

0

Checkbox

☑ Value 1

☑ Value 2

□ Value 3

Date:

2014-01-15

← January 2014 →

Su	Mo	Tu	We	Th	Fr	Sa
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1
2	3	4	5	6	7	8

Illustrating outputs

You entered

```
[1] 0
```

You entered

```
[1] "1" "2"
```

You entered

```
[1] "2014-01-15"
```

Details

- Code that you put before shinyServer in the Server.R function gets called once when you do runApp().
- Code inside the unnamed function of shinyServer(function(input, output)), but not in a reactive statement will run once for every new user (or page refresh).
- Code in reactive functions of shinyServer get run repeatedly as needed when new values are entered.

Argomenti

Shiny

Basic Example

Examples with some complex operations

Others

Other things Shiny can do

- Allow users to upload or download files.
- Have editable data tables.
- Have a dynamic UI.
- User defined inputs and outputs.
- Put a submit button so that Shiny only executes complex code after user hits submit.

Distributing a Shiny Application

- The quickest way is to send (or put on bitbucket or dropbox or whatever) someone the application directory and they can then call runApp.
- You could create an R package and create a wrapper that calls runApp.
- Of course, these solutions only work - if the user knows R.

Distributing a Shiny Application

- Another option is to run a shiny server.
- But thi requires setting up a Shiny server `http://www.rstudio.com/shiny/server/`
- Probably easiest if you use one of the virtual machines where they already have Shiny servers running well.
- Setting up a Shiny server is beyond the scope of this class as it involves some amount of linux server administration.