CSE 590
Data Science Fundamentals

Using R with D3

Klaus Mueller

Computer Science Department
Stony Brook University and SUNY Korea

Lecture	Topic	Projects
1	Intro, schedule, and logistics	
2	Data Science components and tasks	
3	Data types	Project #1 out
4	Introduction to R, statistics foundations	
5	Introduction to D3, visual analytics	
6	Data preparation and reduction	
7	Data preparation and reduction	Project #1 due
8	Similarity and distances	Project #2 out
9	Similarity and distances	
10	Cluster analysis	
11	Cluster analysis	
12	Pattern miming	Project #2 due
13	Pattern mining	
14	Outlier analysis	
15	Outlier analysis	Final Project proposal due
16	Classifiers	
17	Midterm	
18	Classifiers	
19	Optimization and model fitting	
20	Optimization and model fitting	
21	Causal modeling	
22	Streaming data	Final Project preliminary report due
23	Text data	
24	Time series data	
25	Graph data	
26	Scalability and data engineering	
27	Data journalism	
	Final project presentation	Final Project slides and final report due

WHAT IS D3.JS?

D3 = Data Driven Documents

JavaScript library for manipulating documents based on data

- frequent tool to support *data journalism* (New York Times)

D3 helps you bring data to life using HTML, SVG, and CSS

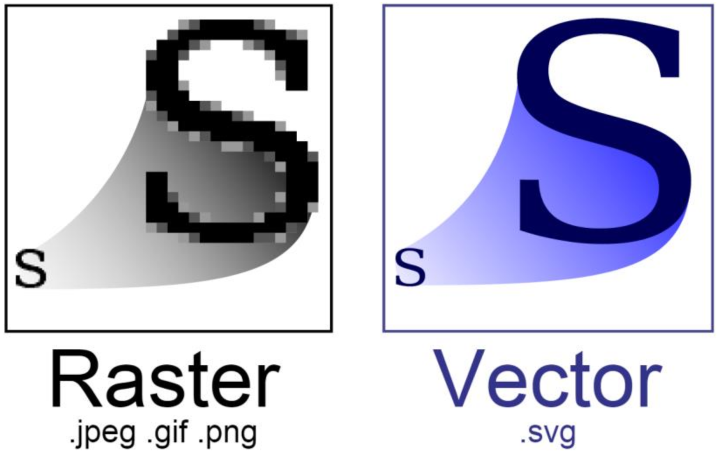
- great library to construct animated visualizations (D3 website)

Runs in any modern web browser (Chrome, Firefox, IE)

- no need to download any software
- independent of OS (Linux, Windows Mac)

Makes Use Of

HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
JS	JavaScript
DOM	The Document Object Model
▪	tree structured organization of HTML objects
SVG	Scalable Vector Graphics



What You Need

A text editor

- textMate, eclipse/aptana, sublime text 2…
- need an editor with syntax highlighting. else it’s easy to get lost

The d3 library

- from http://d3js.org

Data files for your code

A web server (recommended)

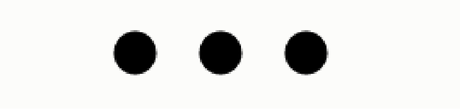
- if your visualization is reading data from files or a database (XMLHttpRequest)
- many options: EasyPHP (windows), Mac OS X Server, MAMP
- else need to specify the data in the code

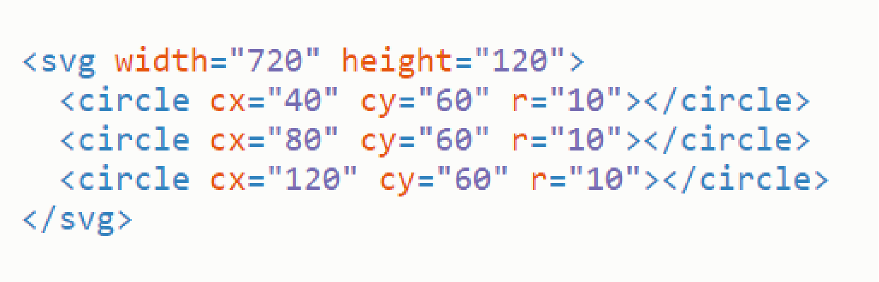
A browser

- to run the code

Selections with D3

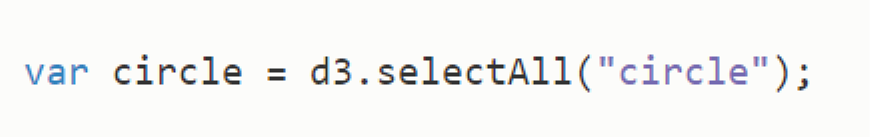
Suppose you defined three circles





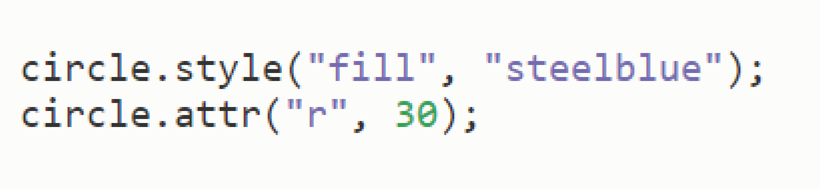
```
<svg width="720" height="120">
  <circle cx="40" cy="60" r="10"></circle>
  <circle cx="80" cy="60" r="10"></circle>
  <circle cx="120" cy="60" r="10"></circle>
</svg>
```

This will select all circles

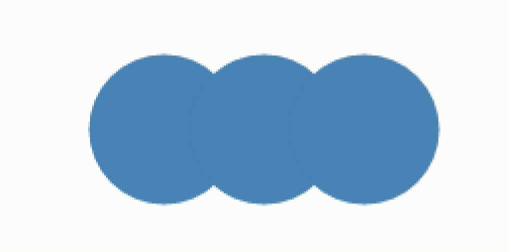


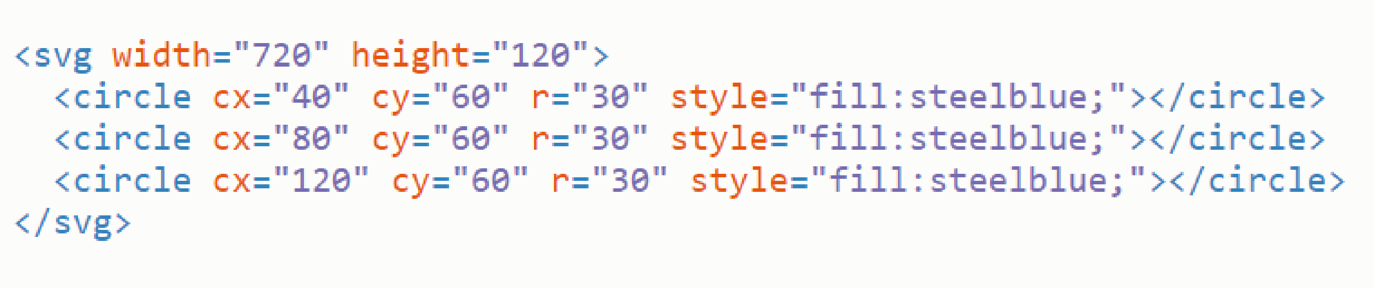
```
var circle = d3.selectAll("circle");
```

And enlarge and fill them



```
circle.style("fill", "steelblue");
circle.attr("r", 30);
```

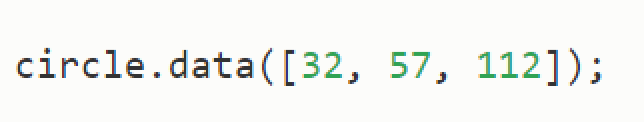




```
<svg width="720" height="120">
  <circle cx="40" cy="60" r="30" style="fill:steelblue;"></circle>
  <circle cx="80" cy="60" r="30" style="fill:steelblue;"></circle>
  <circle cx="120" cy="60" r="30" style="fill:steelblue;"></circle>
</svg>
```

Binding Data to Graphics

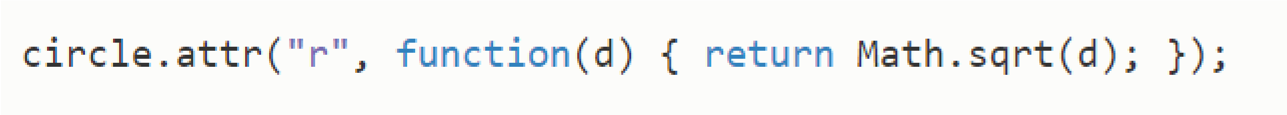
The selection.data method binds the numbers to the circles:



```
circle.data([32, 57, 112]);
```

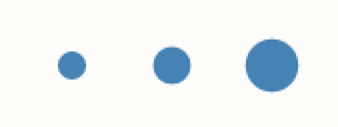
Assign attributes to the bound data

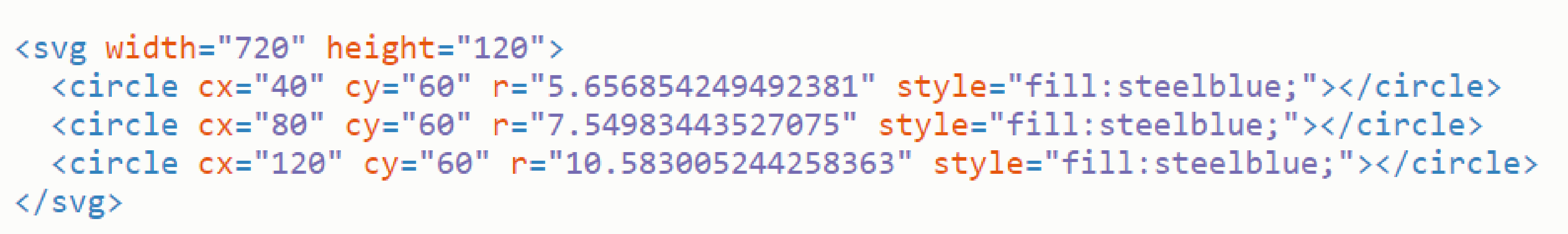
- typically use the name *d* to refer to bound data



```
circle.attr("r", function(d) { return Math.sqrt(d); });
```

Will result in:



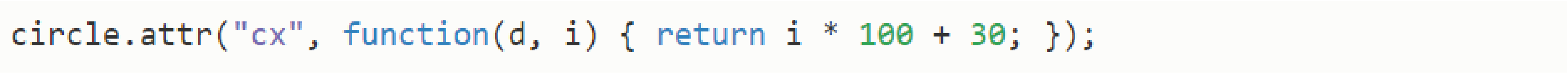


```
<svg width="720" height="120">
  <circle cx="40" cy="60" r="5.656854249492381" style="fill:steelblue;"></circle>
  <circle cx="80" cy="60" r="7.54983443527075" style="fill:steelblue;"></circle>
  <circle cx="120" cy="60" r="10.583005244258363" style="fill:steelblue;"></circle>
</svg>
```

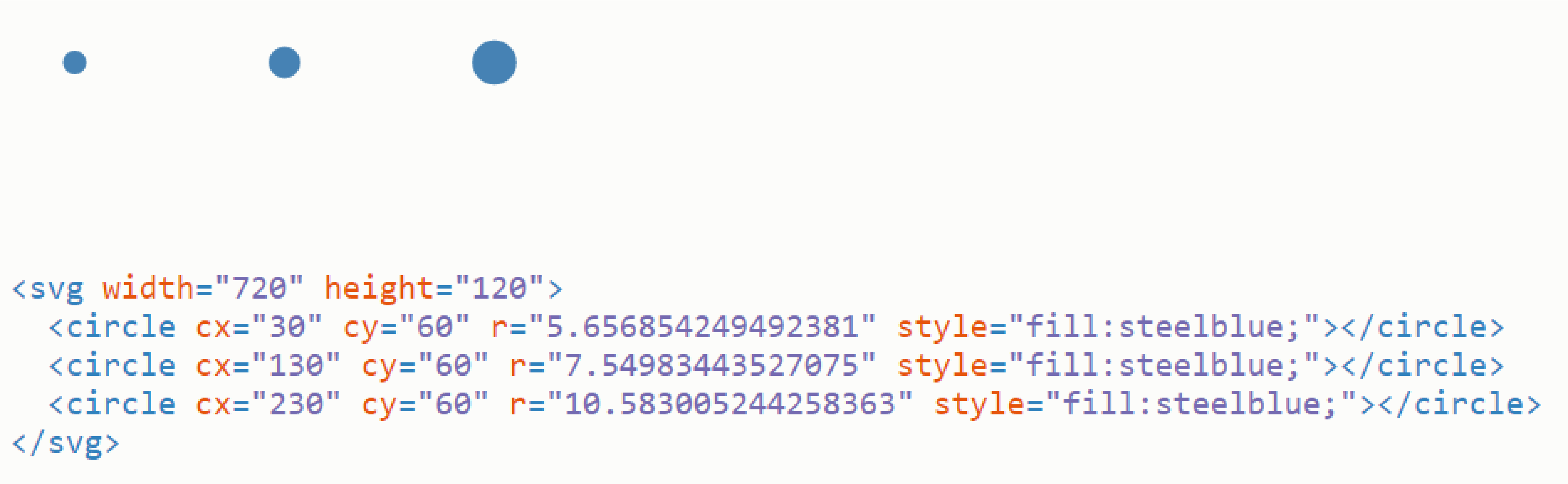
More on Binding Data

We can use the index *i* of the data to define the graphics

Origin is the upper left corner



```
circle.attr("cx", function(d, i) { return i * 100 + 30; });
```

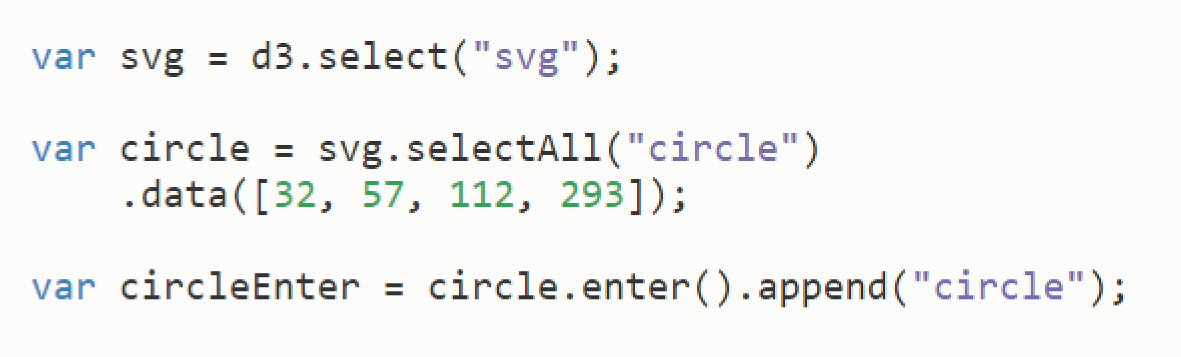


```
<svg width="720" height="120">
  <circle cx="30" cy="60" r="5.656854249492381" style="fill:steelblue;"></circle>
  <circle cx="130" cy="60" r="7.54983443527075" style="fill:steelblue;"></circle>
  <circle cx="230" cy="60" r="10.583005244258363" style="fill:steelblue;"></circle>
</svg>
```

Appending Graphics to Data

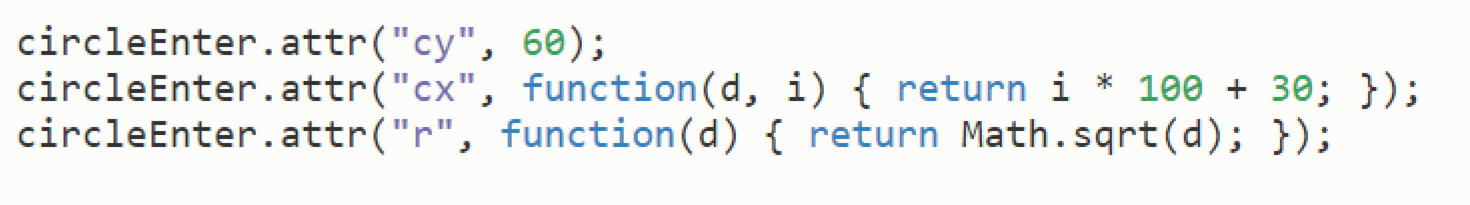
Suppose you have more data than graphics elements

- use the enter method to add them on the fly



```
var svg = d3.select("svg");

var circle = svg.selectAll("circle")
    .data([32, 57, 112, 293]);

var circleEnter = circle.enter().append("circle");
```

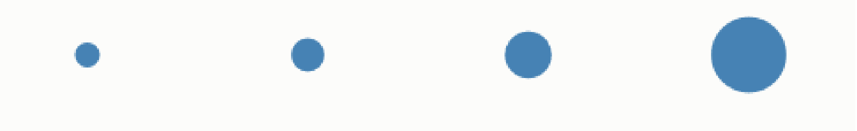
- as usual, but now with 4 circles

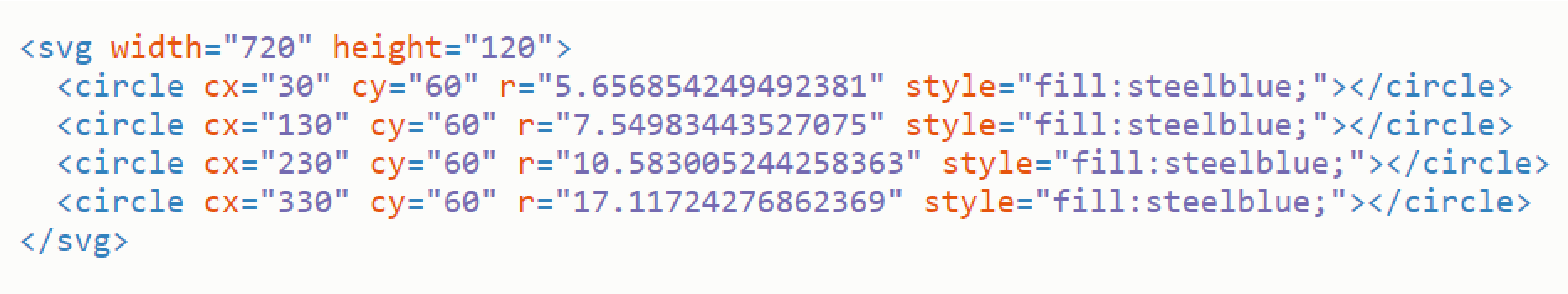


```
circleEnter.attr("cy", 60);
circleEnter.attr("cx", function(d, i) { return i * 100 + 30; });
circleEnter.attr("r", function(d) { return Math.sqrt(d); });
```

Appending Graphics to Data

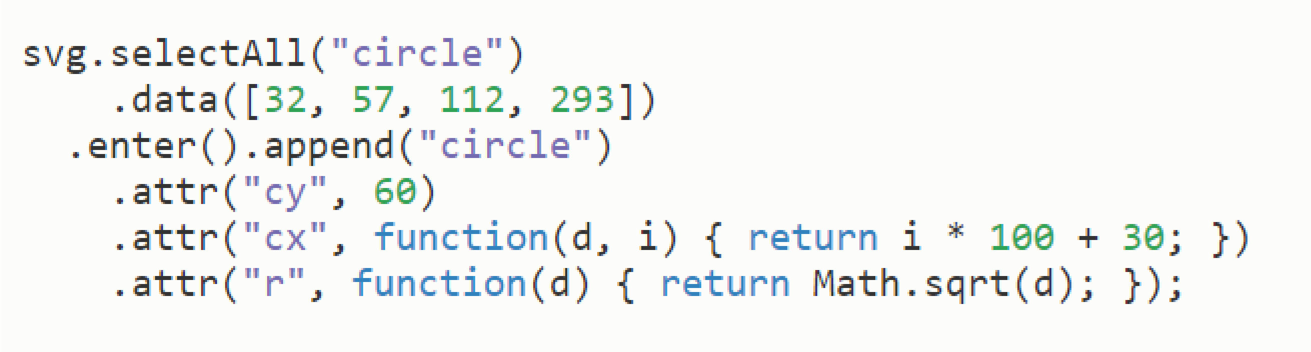
(continued) we get





```
<svg width="720" height="120">
  <circle cx="30" cy="60" r="5.656854249492381" style="fill:steelblue;"></circle>
  <circle cx="130" cy="60" r="7.54983443527075" style="fill:steelblue;"></circle>
  <circle cx="230" cy="60" r="10.583005244258363" style="fill:steelblue;"></circle>
  <circle cx="330" cy="60" r="17.11724276862369" style="fill:steelblue;"></circle>
</svg>
```

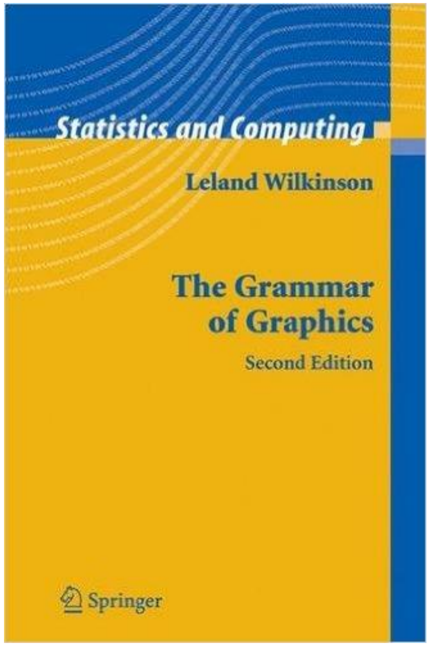
We can even begin with no circles at all:



```
svg.selectAll("circle")
    .data([32, 57, 112, 293])
  .enter().append("circle")
    .attr("cy", 60)
    .attr("cx", function(d, i) { return i * 100 + 30; })
    .attr("r", function(d) { return Math.sqrt(d); });
```

Connecting R with D3

Plotting With R



Simple option: use ggplot2

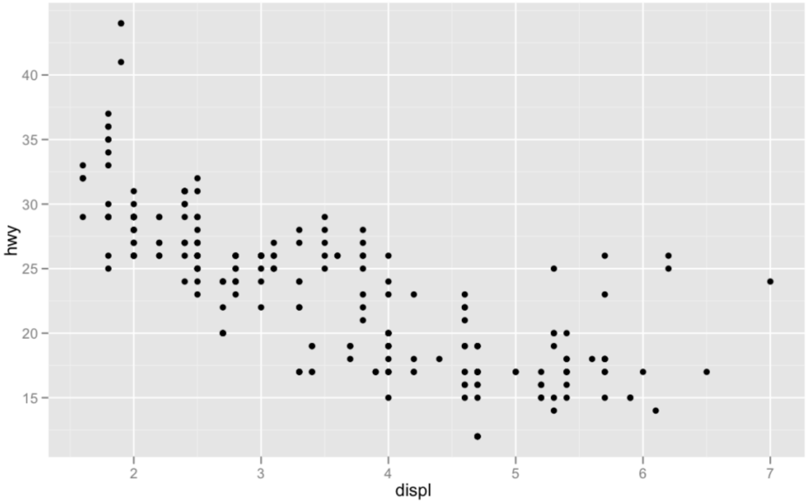
- tries to adhere to the **g**rammar of **g**raphics
- **gg**plot2

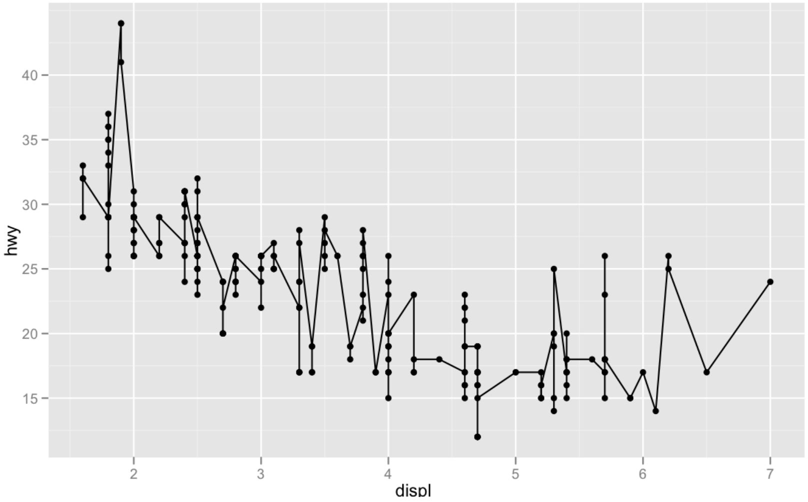
The grammar of graphics

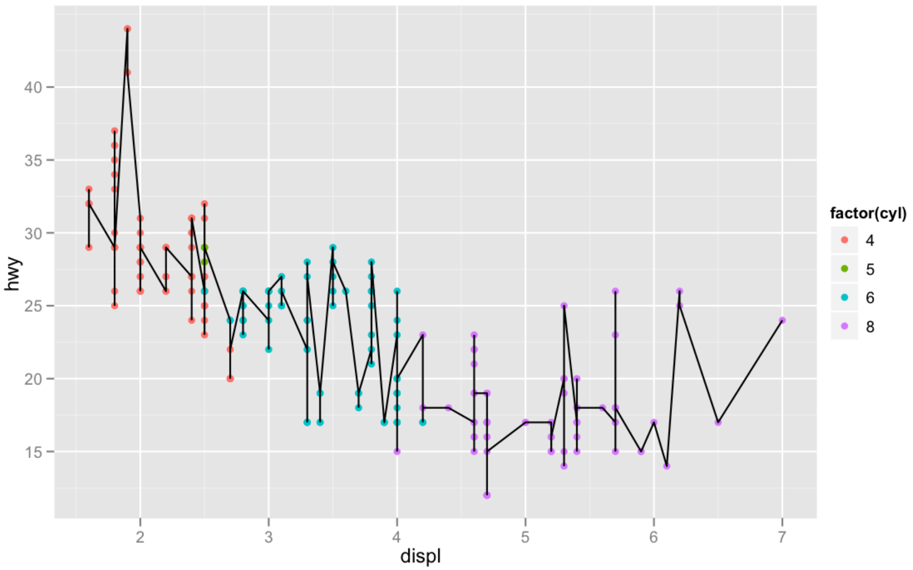
- governs the composition of graphical components in statistical graphics
- by directly controlling that grammar, you can generate a large set of carefully constructed graphics tailored to your particular needs
- controls positions, shapes, appearance, etc. of the primitives
- each component is added to the plot as a layer

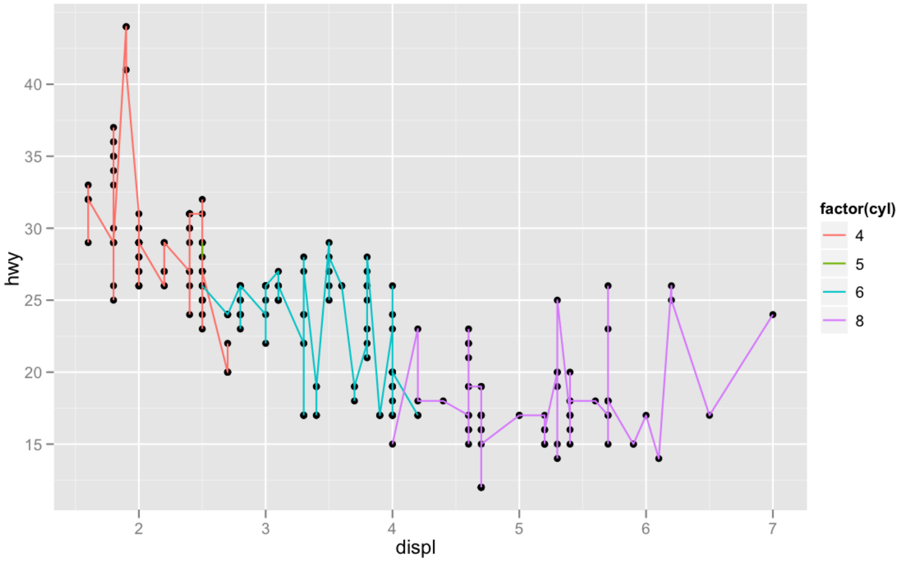
Produces a static plot

LAYERING WITH GGPLOT2









Code For Plots on Last Slide

UL: ggplot(mpg, aes(displ, hwy))+geom_point()

- mpg is the dataset, aes is aesthetic mapping
- geom_point adds the points

UR: ggplot(mpg, aes(displ, hwy))+ geom_point()+geom_line()

- geom_line adds the lines

LL: ggplot(mpg, aes(displ, hwy))+geom_point(aes(color = factor(cyl)))+geom_line()

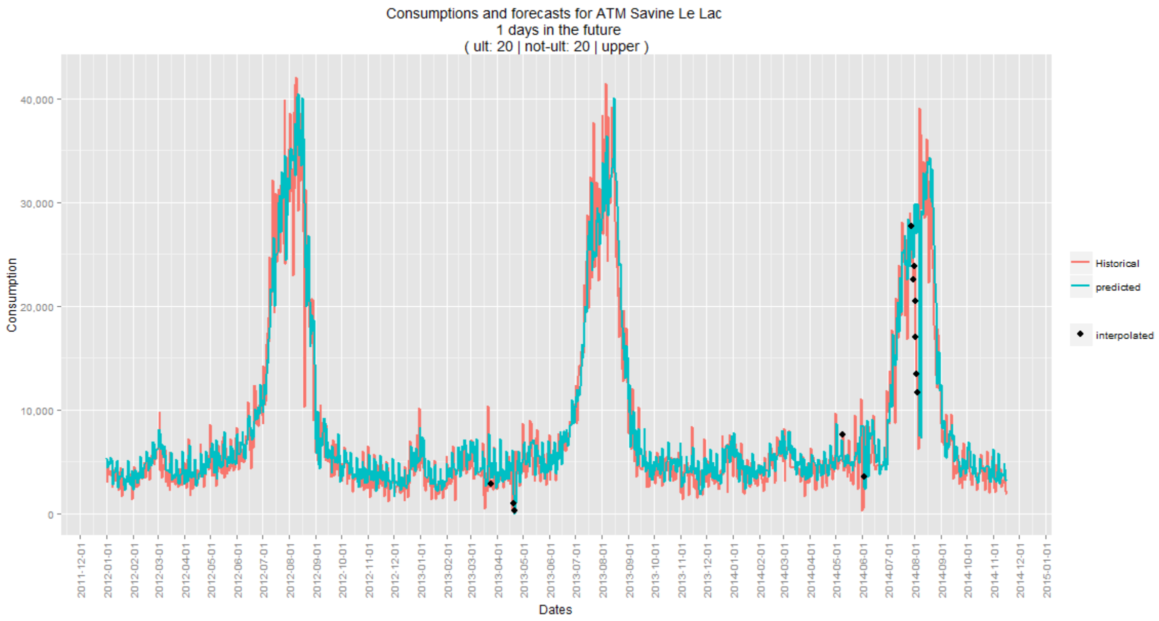
- colored the points by cylinder and auto-added legend

LR: ggplot(mpg, aes(displ, hwy, color = factor(cyl)))+geom_point()+ geom_line()

- colored everything by cylinder and auto-added legend

For more info see here

Ggplot Pros and Cons



Pros

- easy to code
- get an image fast

Cons

- it's a static png image
- can't zoom to see more detail
- need to run the R script again and render a new image

Combining R and D3

Option #1:

- R does the data processing **and** the graph rendering
- then exports this as an SVG
- bind your JavaScript later on
- for 1-way communication from R to a webpage with JavaScript

Option #2:

- R does the data processing
- then sends this data to JavaScript to create an SVGimage
- also enables 2-way communication → rerun R scripts based on input from a web application

Option 1: First plotting, then binding

gridSVG

- requires a lot of manual coding
- example

plotly

- commercial
- also works for Python, Matlab, NodeJS and Excel
- free part has basic, but somewhat limited functionality
- cannot use the full spectrum of the D3 library
- only suitable for 'basic' charts and plots.
- example

Option 2: First binding, then plotting

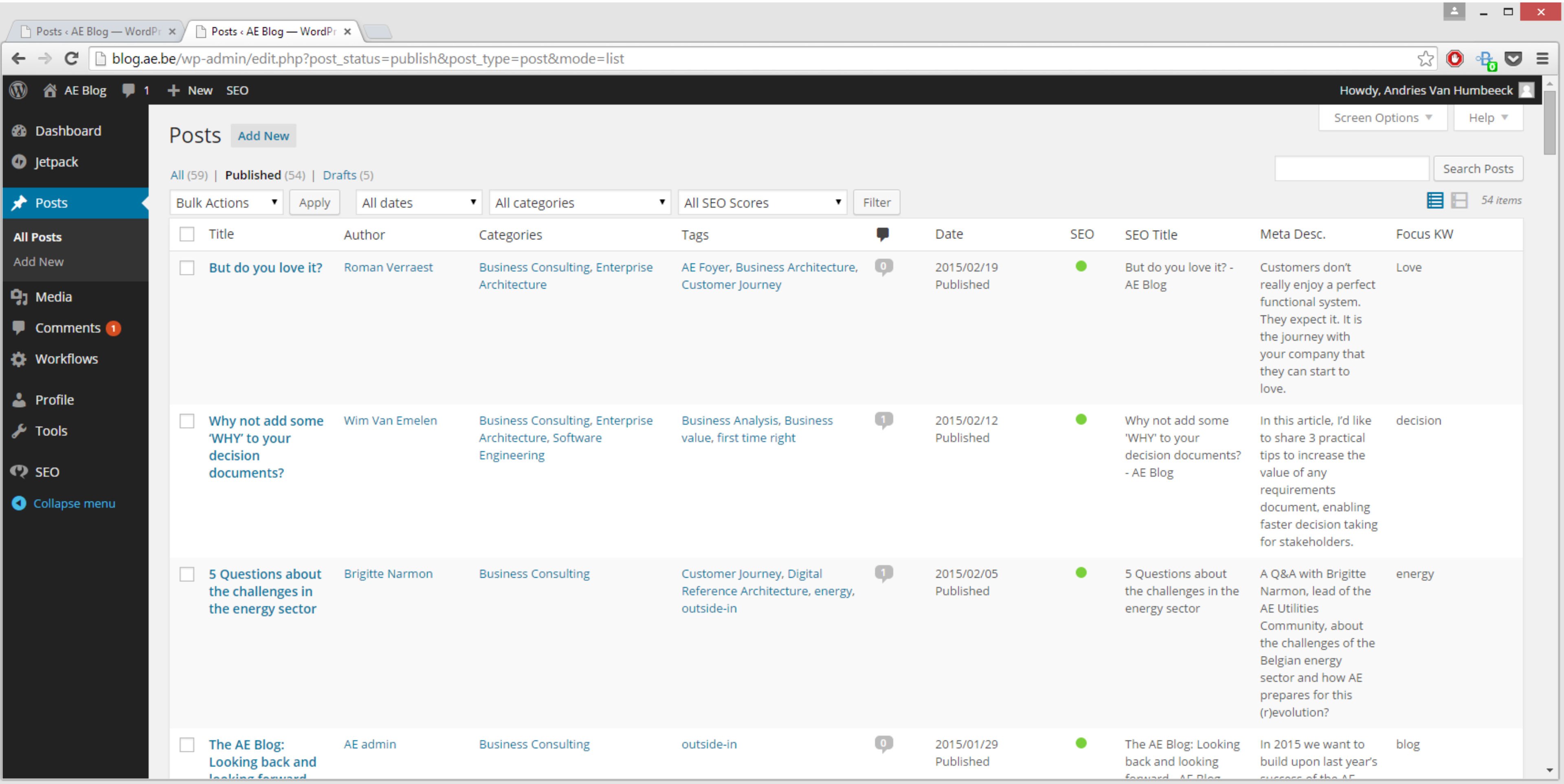
R does the processing and delivers the data

- this is then used as input for the JavaScript visualization

Example

- use R to get and clean the data
- use JS to visualize the data

Example: Blog Site Data



blog.ae.be/wp-admin/edit.php?post_status=publish&post_type=post&mode=list

Posts

All (59) | **Published** (54) | Drafts (5)

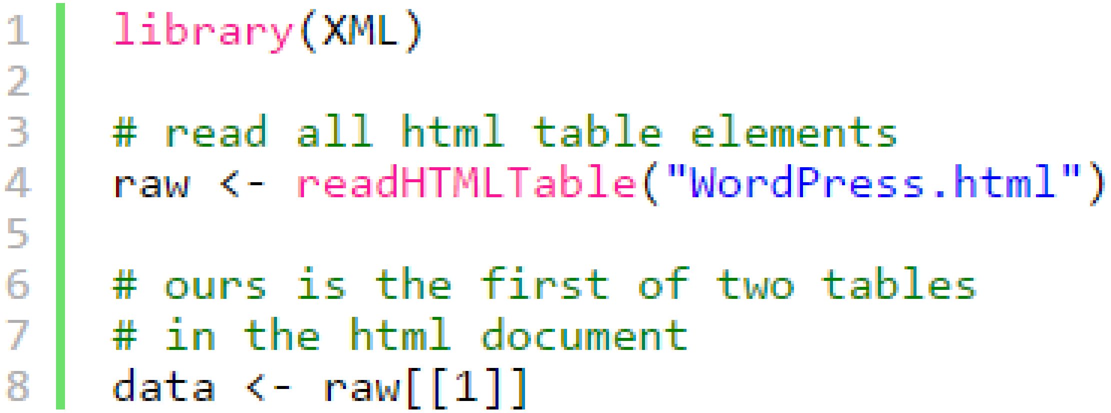
□	Title	Author	Categories	Tags	Comments	Date	SEO	SEO Title	Meta Desc.	Focus KW
□	But do you love it?	Roman Verraest	Business Consulting, Enterprise Architecture	AE Foyer, Business Architecture, Customer Journey	0	2015/02/19 Published	●	But do you love it? - AE Blog	Customers don't really enjoy a perfect functional system. They expect it. It is the journey with your company that they can start to love.	Love
□	Why not add some 'WHY' to your decision documents?	Wim Van Emelen	Business Consulting, Enterprise Architecture, Software Engineering	Business Analysis, Business value, first time right	1	2015/02/12 Published	●	Why not add some 'WHY' to your decision documents? - AE Blog	In this article, I'd like to share 3 practical tips to increase the value of any requirements document, enabling faster decision taking for stakeholders.	decision
□	5 Questions about the challenges in the energy sector	Brigitte Narmon	Business Consulting	Customer Journey, Digital Reference Architecture, energy, outside-in	1	2015/02/05 Published	●	5 Questions about the challenges in the energy sector	A Q&A with Brigitte Narmon, lead of the AE Utilities Community, about the challenges of the Belgian energy sector and how AE prepares for this (r)evolution?	energy
□	The AE Blog: Looking back and looking forward	AE admin	Business Consulting	outside-in	0	2015/01/29 Published	●	The AE Blog: Looking back and looking forward - AE Blog	In 2015 we want to build upon last year's success of the AE	blog

Step #1: Get The Data

Steps:

- the html page contains an overview of all blog posts
- blog-data in this webpage is structured in an html-table
- save this page as a static html page so it can be parse more easily
- R has packages to easily scrape the data from such a table

R code:

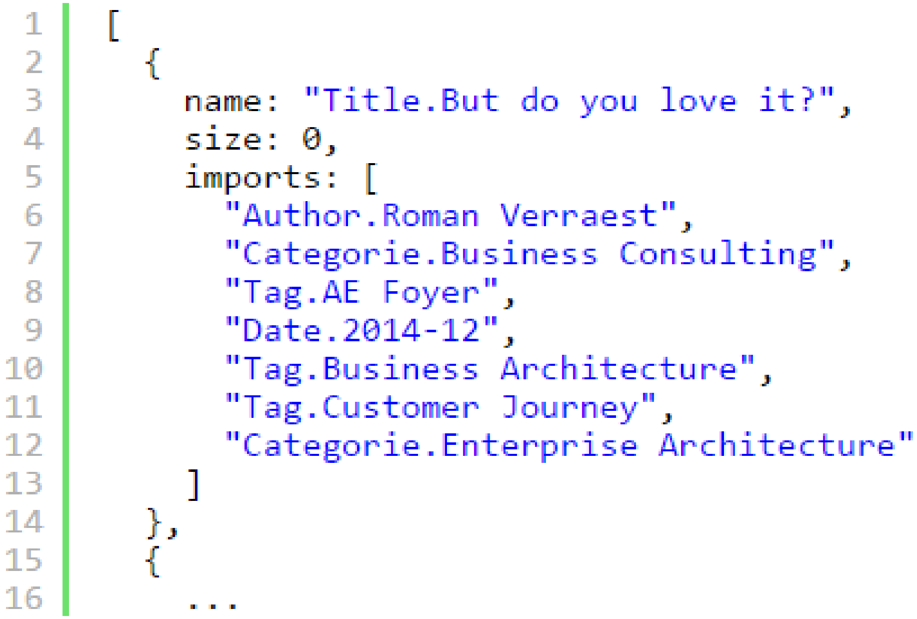


```r
library(XML)

# read all html table elements
raw <- readHTMLTable("WordPress.html")

# ours is the first of two tables
# in the html document
data <- raw[[1]]
```

Step #2: Clean The Data

Data Structure:

- need to store the data in the right format – JSON
- JSON = JavaScript Object Notation, see here
- contains all relations (name/value pairs), for each single element
- aka object, record, struct, dictionary, keyed list, assoc. array, etc.
- will be the data-input for the visualization

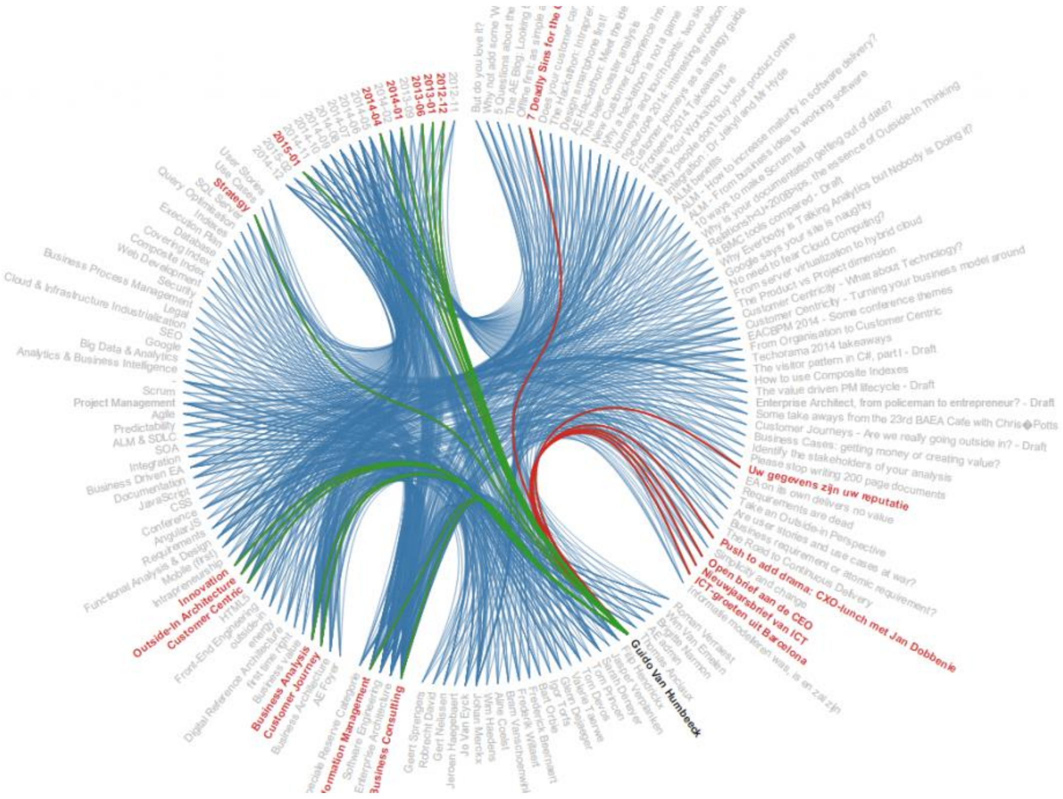


```
[
  {
    name: "Title.But do you love it?",
    size: 0,
    imports: [
      "Author.Roman Verraest",
      "Categorie.Business Consulting",
      "Tag.AE Foyer",
      "Date.2014-12",
      "Tag.Business Architecture",
      "Tag.Customer Journey",
      "Categorie.Enterprise Architecture"
    ]
  },
  {
    ...
```

Step #3: Visualize the data

Pipe into D3

- plug the data into this code will produce this
- move the mouse to see the relations among the different entities.



2012-11
2012-12
2013-01
2013-06
2013-09
2014-01
2014-02
2014-04
2014-05
2014-06
2014-07
2014-08
2014-09
2014-10
2014-11
2015-01
2015-02
2014-12
But do you love it?
Why not add some 'W
5 Questions about the
The AE Blog: Looking
Offline first: as simple a
7 Deadly Sins for the
Does your customer car
The Hackathon: Intrapre
Design smartphone first!
AE Hackathon: Meet the ide
The beer coaster analysis
New Customer Experience Ins
Why a hackathon is not a game
Journeys and touchpoints: two sid
ng-europe 2014: interesting evolution
Customer journeys as a strategy guide
Fronteers 2014 Takeaways
Make Your Workshop Live
Why people don't buy your product online
Integration : Dr Jekyll and Mr Hyde
ALM benefits
ALM - How to increase maturity in software delivery?
ALM - From business idea to working software
10 ways to make Scrum fail
Why is your documentation getting out of date?
Relationsh<U+2008>ips, the essence of Outside-In Thinking
4 BMC tools compared - Draft
Why Everbody is Talking Analytics but Nobody is Doing it?
Google says your site is naughty
No need to fear Cloud Computing?
From server virtualization to hybrid cloud
The Product vs Project dimension
Customer Centricity - What about Technology?
Customer Centricity - Turning your business model around
EACBPM 2014 - Some conference themes
From Organisation to Customer Centric
Techorama 2014 takeaways
The visitor pattern in C#, part I - Draft
How to use Composite Indexes
The value driven PM lifecycle - Draft
Enterprise Architect, from policeman to entrepreneur? - Draft
Some take aways from the 23rd BAEA Cafe with Chris�Potts
Customer Journeys - Are we really going outside in? - Draft
Business Cases: getting money or creating value?
Identify the stakeholders of your analysis
Please stop writing 200 page documents
Uw gegevens zijn uw reputatie
EA on its own delivers no value
Requirements are dead
Take an Outside-in Perspective
Are user stories and use cases at war?
Business requirement or atomic requirement?
The Road to Continuous Delivery
Push to add drama: CXO-lunch met Jan Dobbenie
Simplicity and change
Open brief aan de CEO
Nieuwjaarsbrief van ICT
ICT-groeten uit Barcelona
Informatie modelleren was, is en zal zijn
Roman Verraest
Wim Van Emelen
Brigitte Narmon
AE admin
Thomas Anciaux
Guido Van Humbeeck
Filip Hendrickx
Jasper Verplanken
Sarah Demeyer
Tom Princen
Tom Devos
Valerie Taerwe
Glenn Dejaeger
Igor Torfs
Bart Orbie
Frederick Beernaert
Frederik Willaert
Bram Vanschoenwinkel
Aline Coelst
Wim Haeders
Johan Merckx
Jo Van Eyck
Jeroen Haegebaert
Gert Nelissen
Robrecht David
Geert Sprengers
Business Consulting
Enterprise Architecture
Software Engineering
Information Management
ecide Reserve Categorie
AE Foyer
Business Architecture
Customer Journey
Business Analysis
Business value
first time right
Digital Reference Architecture
energy
outside-in
Front-End Engineering
HTML5
Customer Centric
Outside-In Architecture
Innovation
Intrapreneurship
Mobile (first)
Functional Analysis & Design
Requirements
AngularJS
Conference
CSS
JavaScript
Documentation
Business Driven EA
Integration
SOA
ALM & SDLC
Predictability
Agile
Project Management
Scrum
-
Analytics & Business Intelligence
Big Data & Analytics
Google
SEO
Cloud & Infrastructure Industrialization
Legal
Business Process Management
Security
Web Development
Composite Index
Covering Index
Database
Execution Plan
Indexes
Query Optimization
SQL Server
Strategy
Use Cases
User Stories

Two-Way Communication

Rerun R scripts based on input from a web application

- this is a bit more involved using D3 – not covered in this lecture
- may not be needed for this course
- Shiny webserver can do this

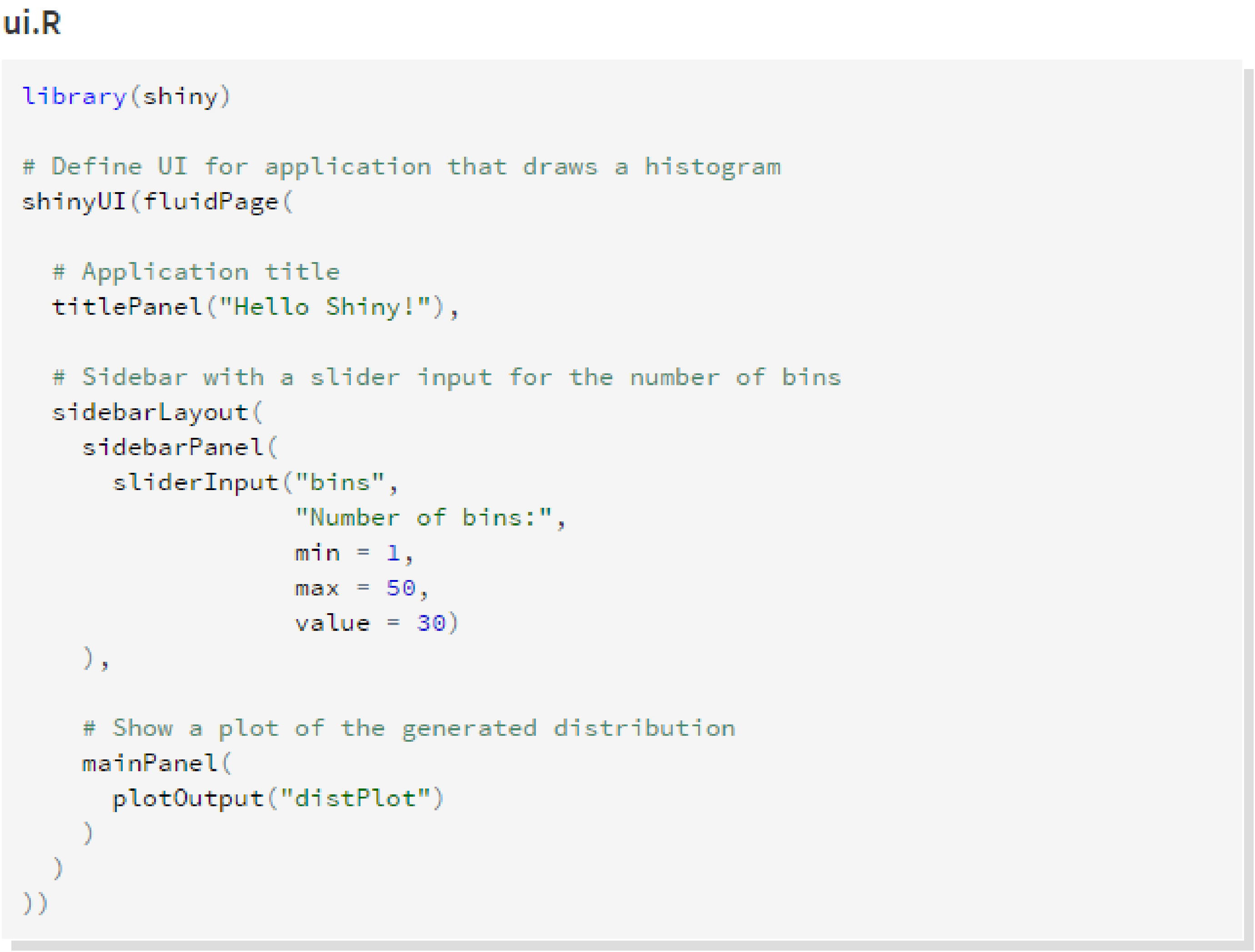
Shiny

- R package that makes it easy to build interactive web apps
- currently free and available here
- this site has also a comprehensive tutorial
- uses D3 but hides it from the user completely
- this can be limiting

Structure of a Shiny App – two components

- a user-interface script
- a server script

Shiny Example – User Interface

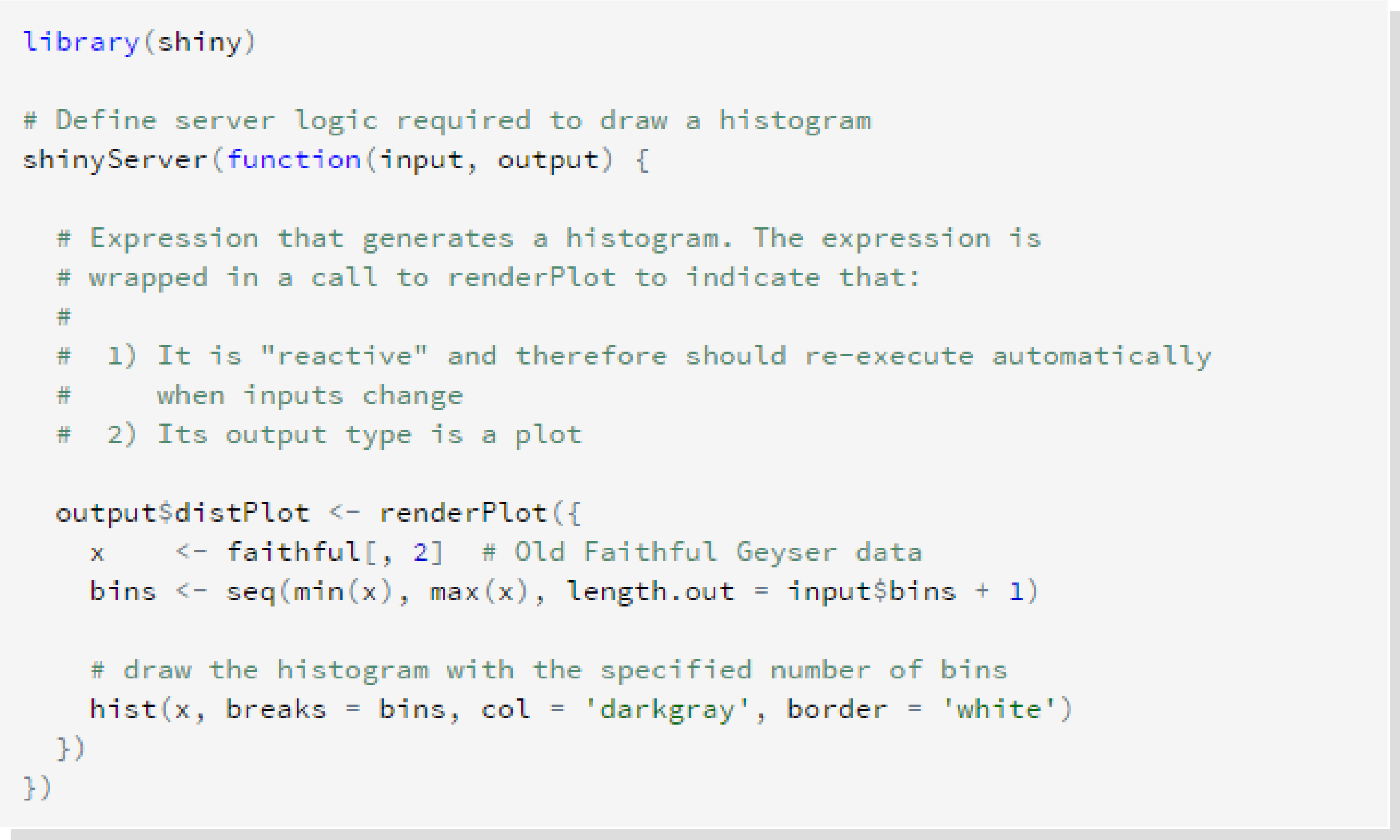


ui.R

```r
library(shiny)

# Define UI for application that draws a histogram
shinyUI(fluidPage(

  # Application title
  titlePanel("Hello Shiny!"),

  # Sidebar with a slider input for the number of bins
  sidebarLayout(
    sidebarPanel(
      sliderInput("bins",
                  "Number of bins:",
                  min = 1,
                  max = 50,
                  value = 30)
    ),

    # Show a plot of the generated distribution
    mainPanel(
      plotOutput("distPlot")
    )
  )
))
```

Shiny Example – Server Script

server.R



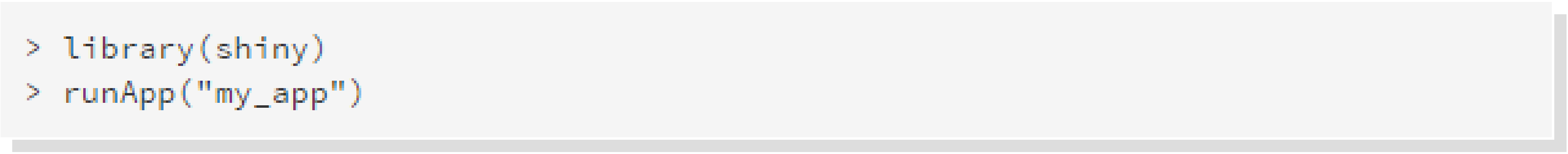
```r
library(shiny)

# Define server logic required to draw a histogram
shinyServer(function(input, output) {

  # Expression that generates a histogram. The expression is
  # wrapped in a call to renderPlot to indicate that:
  #
  #  1) It is "reactive" and therefore should re-execute automatically
  #     when inputs change
  #  2) Its output type is a plot

  output$distPlot <- renderPlot({
    x    <- faithful[, 2]  # Old Faithful Geyser data
    bins <- seq(min(x), max(x), length.out = input$bins + 1)

    # draw the histogram with the specified number of bins
    hist(x, breaks = bins, col = 'darkgray', border = 'white')
  })
})
```

Running Shiny

Note:

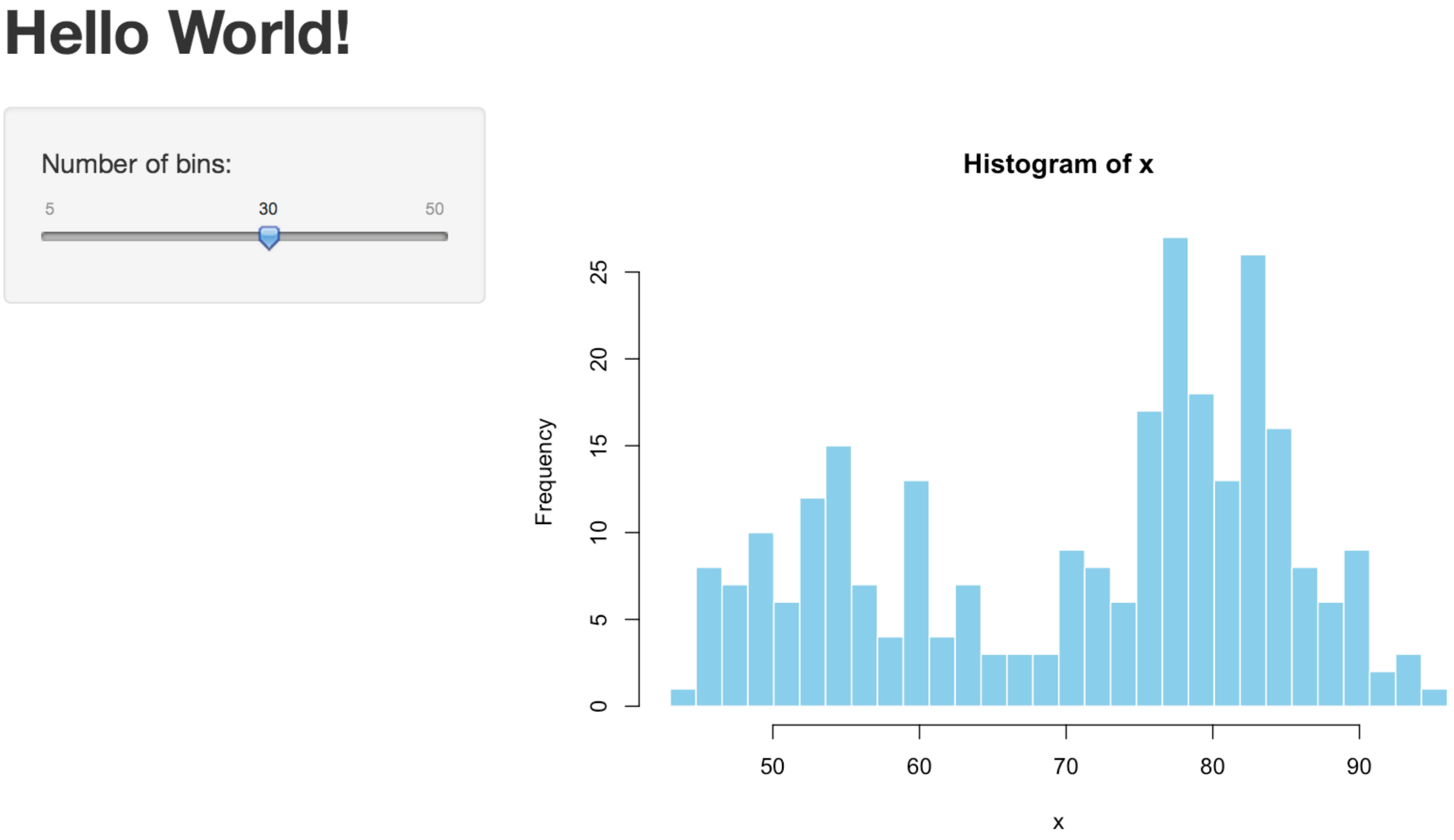
- your R session will be busy while the Hello Shiny app is active
- you will not be able to run any R commands
- R is monitoring the app and executing the app’s reactions
- to get your R session back, hit escape or click the stop sign in the RStudio console panel

Run app by:



```
> library(shiny)
> runApp("my_app")
```

Shiny Example App



Hello World!

Number of bins: 30

More Information

Most of the material presented here was from here

- but there is much more information on the web

What should you choose?

- ggplot2 (easy)
- shiny (a bit more difficult, but still easy)
- binding in D3 (takes a learning curve, but rewarding)
- up to you – get as much as you want out of this course

- (I would choose D3)

More Reading on D3

The page where the D3 tutorial bits came from:
http://www.lessonpaths.com/learn/i/begin-with-d3js/d3js-simplest-examples-of-d3js

Another good tutorial
http://alignedleft.com/tutorials/d3

Now to a more detailed, but still primitive example:
http://www.lessonpaths.com/learn/i/begin-with-d3js/d3js-simplest-examples-of-d3js

Here are some full-fledged implementations:
https://github.com/mbostock/d3/wiki/Gallery