Creative Interaction Design and Technologies
TANG, Xuetong

Data visualization

Agenda

Lecture

- Data sources: Crowed-sourced data & IoT
- Data visualization
- Graph types and guidelines
- How to choose the most appropriate graphic forms
- Personal web development assignment review

Lab Time

- Continuing with Python basics
- Introduction to Pandas
- Introduction to Beautiful Soup
- Web scraping with Beautiful Soup, using IMDB as example

- 宋城薇：https://dhdhsm.github.io/
- 李睿莹：https://liruiying1020.github.io/
- 翁卉：https://nottobeheartbroken.github.io/qizhongzuoye/
- 刘怡鸽：https://stockholmdove.github.io/midtermsite/
- 赵一凡：https://bbbbjo.github.io/bootstrapmidterm/HomeDesign.html
- 刘千瑜：https://jiandanai987.github.io/midterm/
- 袁佳珂：https://littlebeeshines.github.io/
- 宝立格：https://eldenbob.github.io/eat
- https://eldenbob.github.io/vision/welcome
- 杨月：https://yangyue237.github.io/museum
- 陈欣宁：https://18857802804.github.io/bootstrap.html
- 聂旭冉：https://ni-zhann.github.io/

- 跳出率（Bounce Rate）
- 意料之外的：音乐
- 布局与板式为内容服务，而非堆砌
- 轮播图满屏或大于第一屏

Bootstrap template

Google

free bootstrap template

All | Images | News | Videos | Shopping | More | Settings | Tools

About 57,900,000 results (0.47 seconds)

startbootstrap.com › themes

Free Bootstrap Themes & Templates - Start Bootstrap - Start ...

Free themes for **Bootstrap** 4 that are open source, MIT licensed, and **free** to download - these pre-designed **themes** are easy to customize and ready to publish.

Bootstrap Templates · Buy Bootstrap Themes · SB Admin 2 · Freelancer

You've visited this page 2 times. Last visit: 3/27/20

startbootstrap.com

Start Bootstrap: Free Bootstrap Themes, Templates, Snippets ...

Start **Bootstrap** develops **free** to download, open source **Bootstrap** 4 **themes**, **templates**, and snippets and creates guides and tutorials to help you learn more ...

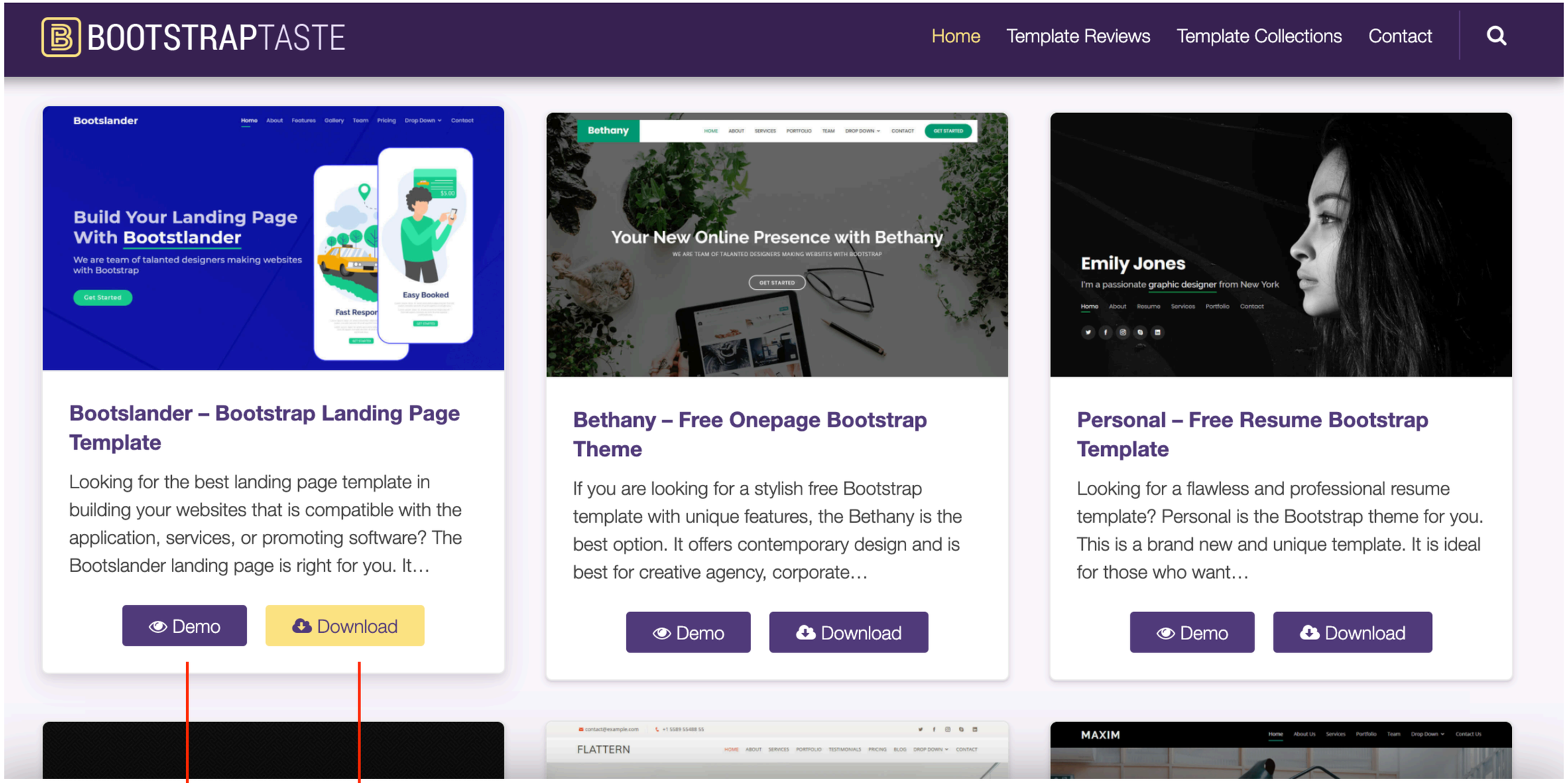
bootstrapmade.com

BootstrapMade: Free Bootstrap Themes and Website Templates

At BootstrapMade, we create beautiful website **templates** and **bootstrap themes** using **Bootstrap**, the most popular HTML, CSS and JavaScript framework.

Bootstrap 4 · Squadfree - Free Bootstrap ... · Appland - Free Bootstrap App ... · Bocor

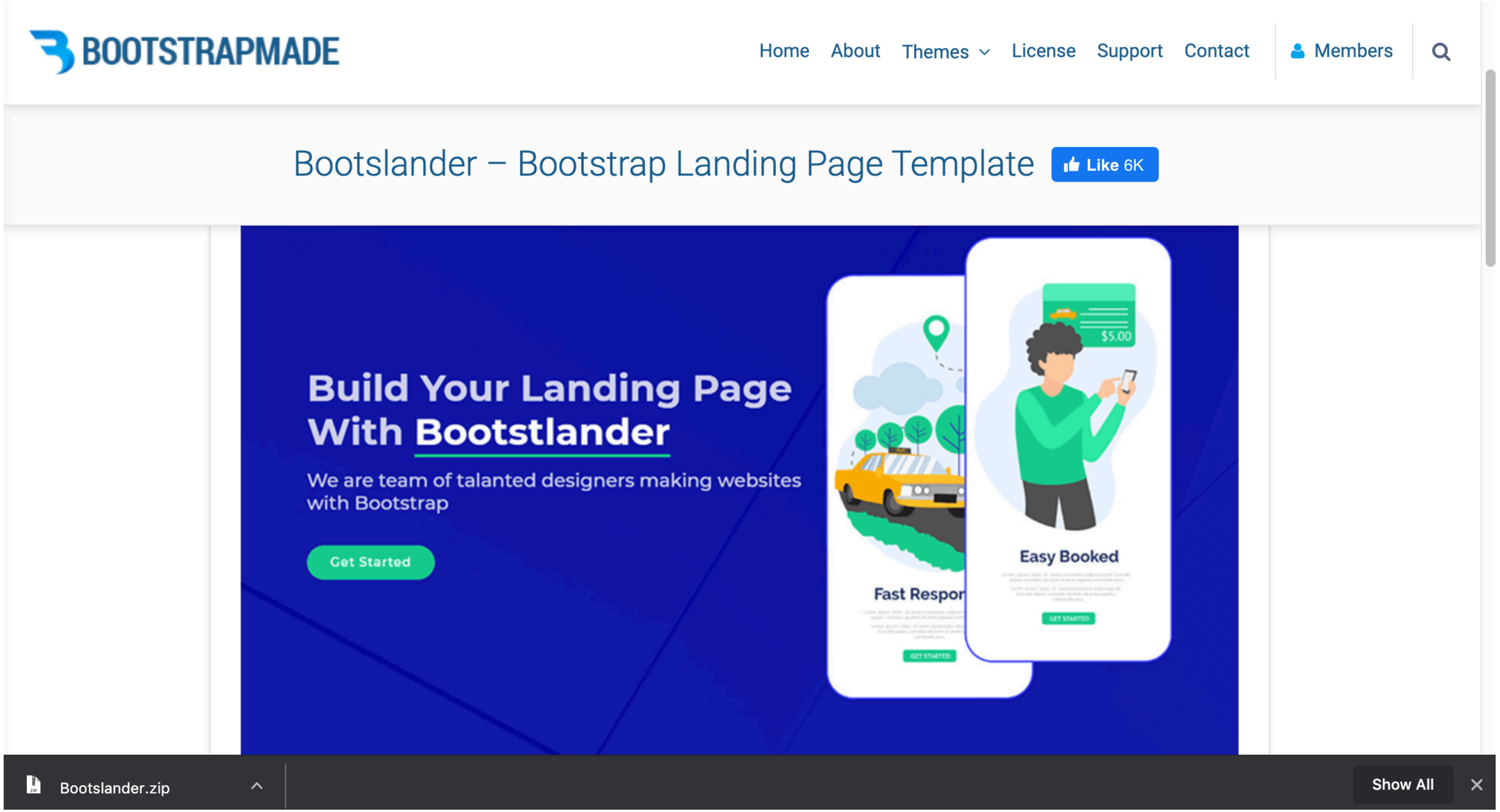
www.creative-tim.com › Bootstrap Themes



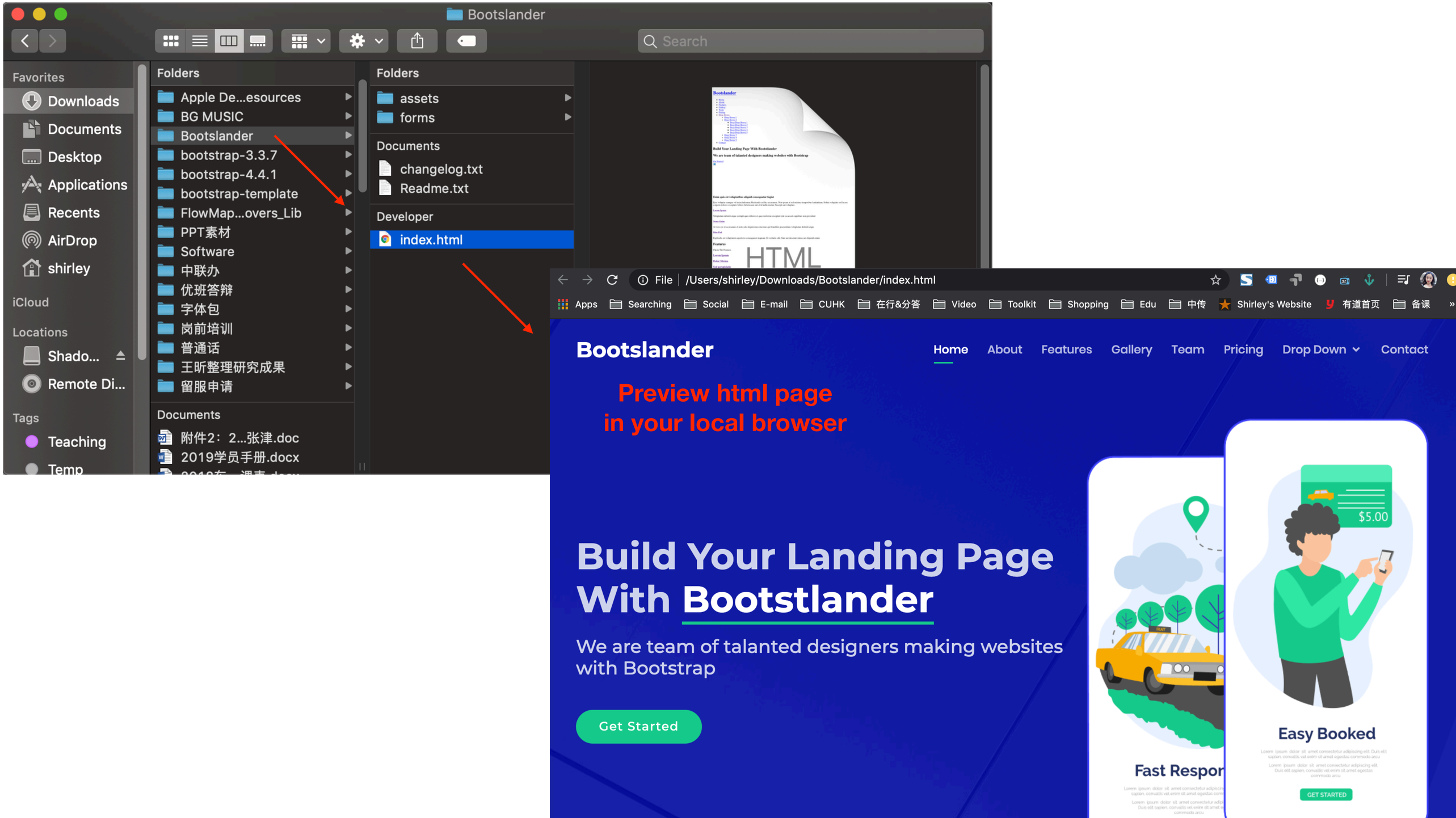
Source: https://bootstraptaste.com/

Preview **Download**

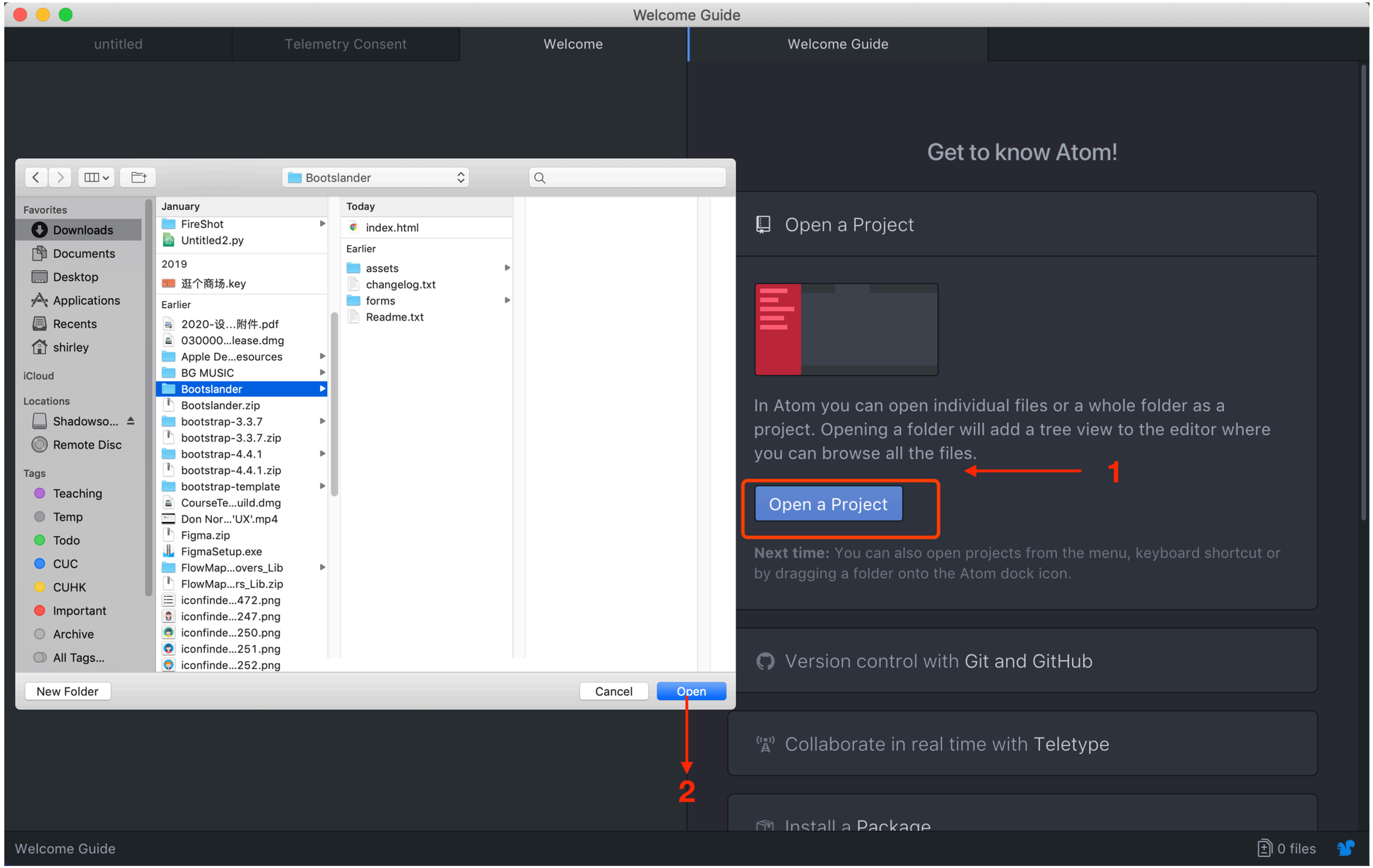
预览 **下载代码&源文件**



Find this file on your local pc and unzip it


Bootslander
Search
Favorites
Downloads
Documents
Desktop
Applications
Recents
AirDrop
shirley
iCloud
Locations
Shado...
Remote Di...
Tags
Teaching
Temp
Folders
Apple De...esources
BG MUSIC
Bootslander
bootstrap-3.3.7
bootstrap-4.4.1
bootstrap-template
FlowMap...overs_Lib
PPT素材
Software
中联办
优班答辩
字体包
岗前培训
普通话
王昕整理研究成果
留服申请
Documents
附件2：2...张津.doc
2019学员手册.docx
Folders
assets
forms
Documents
changelog.txt
Readme.txt
Developer
index.html
HTML
File /Users/shirley/Downloads/Bootslander/index.html
Apps
Searching
Social
E-mail
CUHK
在行&分答
Video
Toolkit
Shopping
Edu
中传
Shirley's Website
有道首页
备课
Bootslander
Home
About
Features
Gallery
Team
Pricing
Drop Down
Contact
Preview html page
in your local browser
Build Your Landing Page
With Bootstlander
We are team of talanted designers making websites
with Bootstrap
Get Started
$5.00
Easy Booked
GET STARTED

Open this file as project in Atom to get its source code



Project — ~/Downloads/Bootslander

Project

- Bootslander
  - assets
  - forms
  - changelog.txt
  - index.html
  - Readme.txt

Now you may change the code or rewrite as you will.

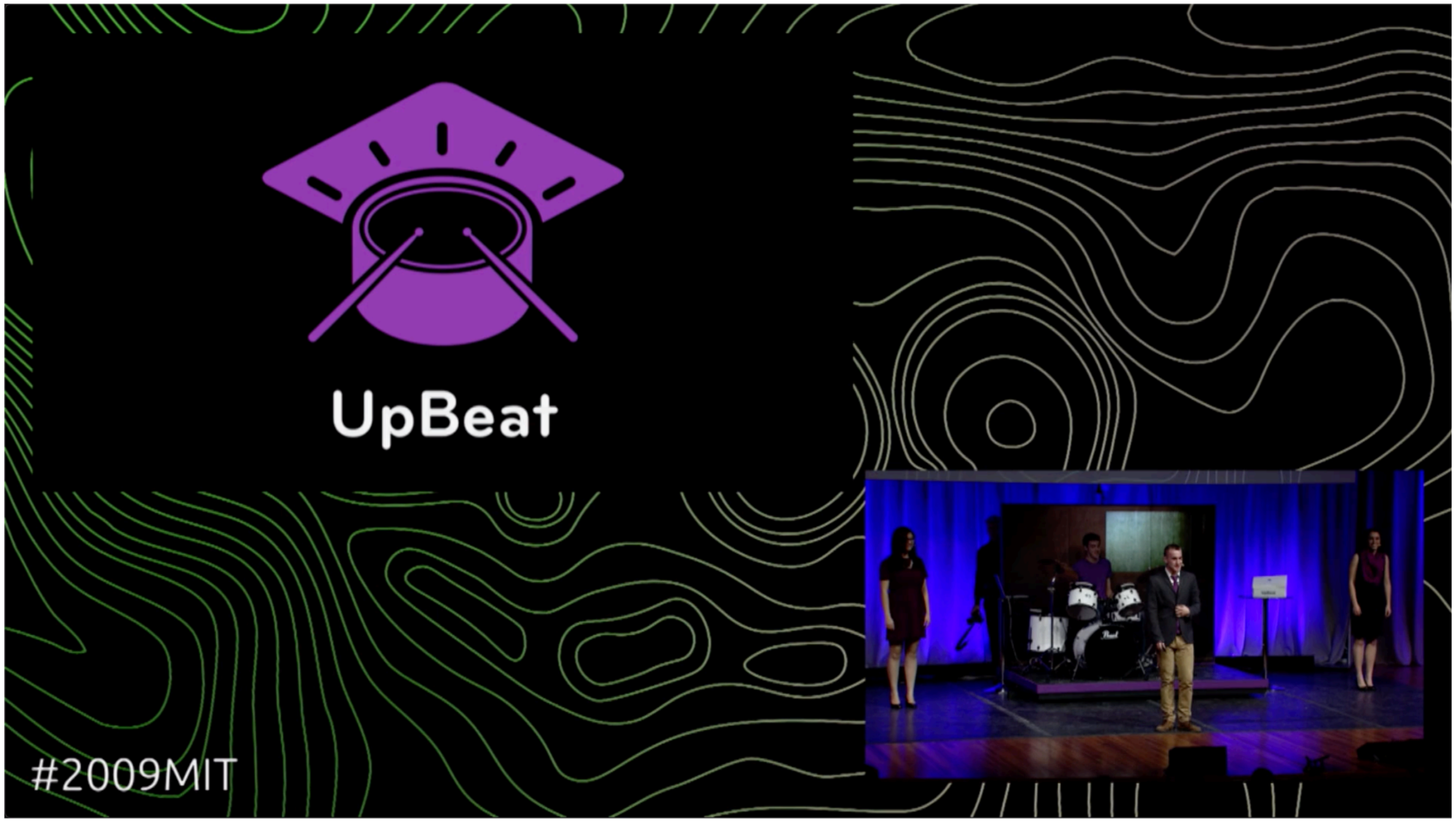
index.html

```html
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta content="width=device-width, initial-scale=1.0" name="viewport">

  <title>Bootslander Bootstrap Template - Index</title>
  <meta content="" name="descriptison">
  <meta content="" name="keywords">

  <!-- Favicons -->
  <link href="assets/img/favicon.png" rel="icon">
  <link href="assets/img/apple-touch-icon.png" rel="apple-touch-icon">

  <!-- Google Fonts -->
  <link href="https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,600,600i,700,700i|Montseri

  <!-- Vendor CSS Files -->
  <link href="assets/vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">
  <link href="assets/vendor/icofont/icofont.min.css" rel="stylesheet">
  <link href="assets/vendor/boxicons/css/boxicons.min.css" rel="stylesheet">
  <link href="assets/vendor/venobox/venobox.css" rel="stylesheet">
  <link href="assets/vendor/remixicon/remixicon.css" rel="stylesheet">
  <link href="assets/vendor/owl.carousel/assets/owl.carousel.min.css" rel="stylesheet">
  <link href="assets/vendor/aos/aos.css" rel="stylesheet">

  <!-- Template Main CSS File -->
  <link href="assets/css/style.css" rel="stylesheet">

  <!-- =======================================================
  * Template Name: Bootslander - v2.1.0
  * Template URL: https://bootstrapmade.com/bootslander-free-bootstrap-landing-page-template/
  * Author: BootstrapMade.com
  * License: https://bootstrapmade.com/license/
```

index.html 1:1 CRLF UTF-8 HTML 0 files

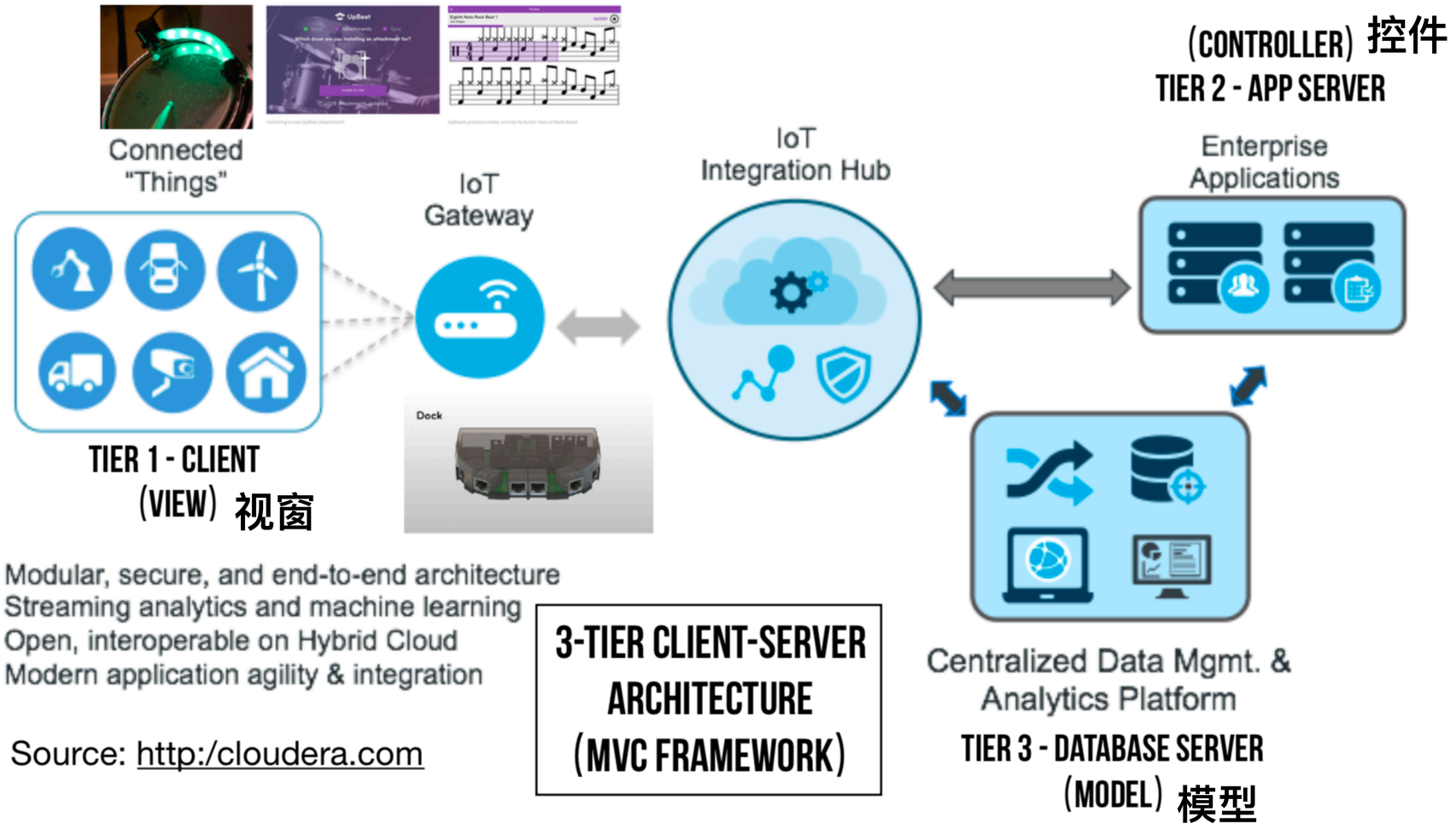
Recap

Case study: UpBeat



Source: https://vimeo.com/114149570

MVC framework



- Modular, secure, and end-to-end architecture
- Streaming analytics and machine learning
- Open, interoperable on Hybrid Cloud
- Modern application agility & integration

Source: http:/cloudera.com

3-TIER CLIENT-SERVER ARCHITECTURE (MVC FRAMEWORK)

Data sources

Open Data

government public databases
international organizations
academic organizations
commercial companies

API

Internal Database

Crowdsourced Data

IOT

Scraping

...

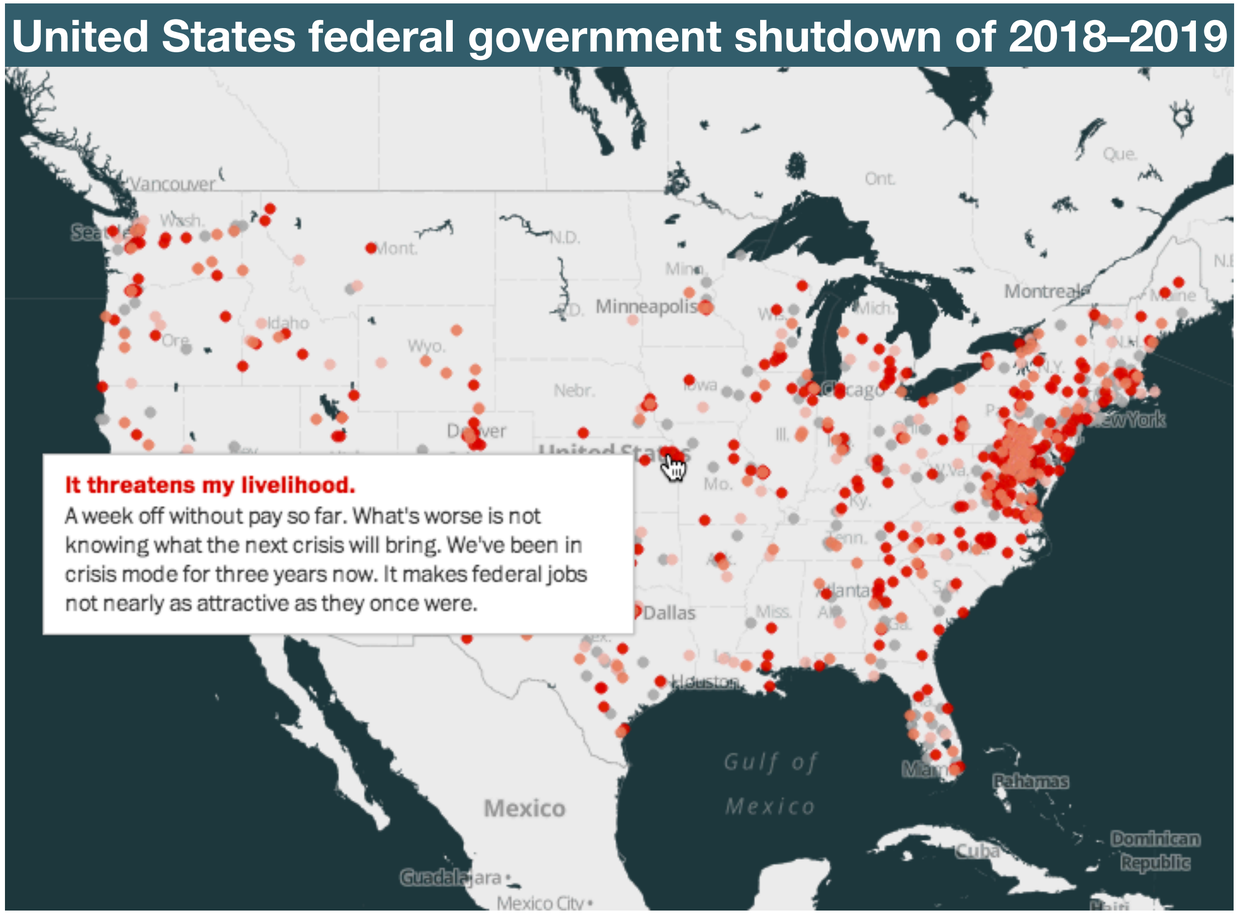
Data sources - crowdsourced data

- Crowdsourcing，按照字面意思来理解：寻求大众的力量来帮忙，把一个工作分解成很多的小分支，交给不同的人来做，再通过收集众人做好的内容并加以整理和发布，成为一个更具意义的内容。
- Crowdsourced data 众包数据，即通过与很多不同的人合作共建数据集，每个人都作为一个节点提供自己的数据，并最终形成一个更具意义的数据集，加以整理后发布，供更多人使用。
- 常见案例：Google Maps、高德地图实时路况、平均车速；Facebook、Instagram、小红书地点打卡；疫情当下，患者同小区、患者同乘等。

Data sources - crowdsourced data

Crowdsourced data 在数据新闻（Data Journalism）中的应用案例

This **government shutdown map by the Washington Post** allows readers to comment how the closed government affects them and shows their response on a MapBox map. The result is an evolving visualization of where in the United States people are affected.



Source: Mapbox

Data sources - crowdsourced data

- Crowdsourced data 在数据新闻（Data Journalism）中的应用案例

The Washington Post
Democracy Dies in Darkness

Politics Primary results White House Con

Politics • Analysis

Mapping where America has been shut down

A customer leaves a McDonald's restaurant with takeout in Brooklyn on Tuesday. (JUSTIN LANE/EPA-EFE/Shutterstock)

By **Philip Bump**

March 17, 2020 at 3:15 PM EDT

At the beginning of the year, it would have

Connecticut came together to set an agreed-upon limit on bars and restaurants. They were joined by Pennsylvania, which restricted "non-essential businesses."

The measures in place to limit the spread of the virus

No measures · Some schools closed · Schools closed · Non-essential closures · Shelter-in-place · Quarantine

What's interesting is that politics doesn't always play a role in the closures. While the governors of Oklahoma and West Virginia have encouraged residents to go out to restaurants, both states have nonetheless shut their schools for the time being in an effort to stop the spread of the virus.

Data sources - IOT

- Internet of Things，物联网，简称IoT，是由物理对象（所谓的“物”）组成的网络，这些物理对象嵌入了传感器、软件和其他技术，以便可以通过互联网与其他设备和系统建立连接并交换数据。
- 物联网设备种类繁多，既有普通家庭用品，也有复杂工业用具。目前，接入互联网的物联网设备已超过 70 亿，专家预计到 2020 年，这一数字将增长到 100 亿，到 2025 年将增长到 220 亿。

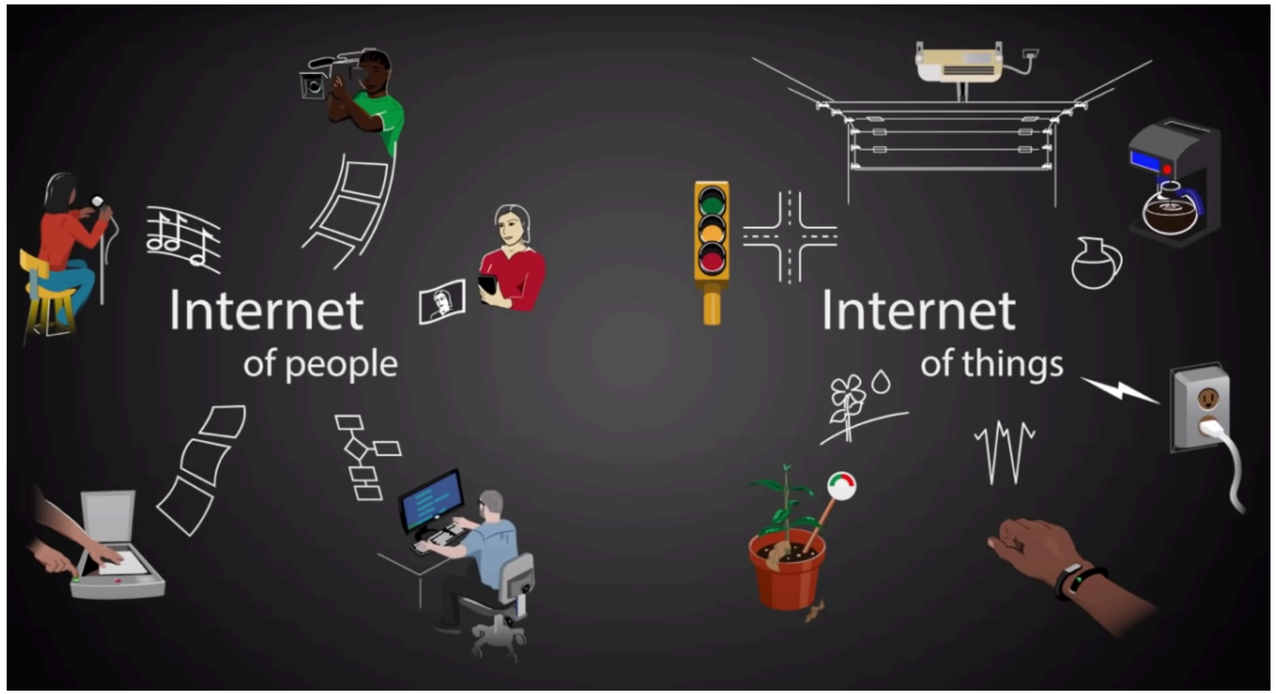
——来自Oracle

Data sources - IOT

物联网（IoT）是什么？看了你就知道

Source: https://www.youtube.com/watch?v=X9KTku6jd6c

Data sources - IOT



Internet of People

Internet of Things

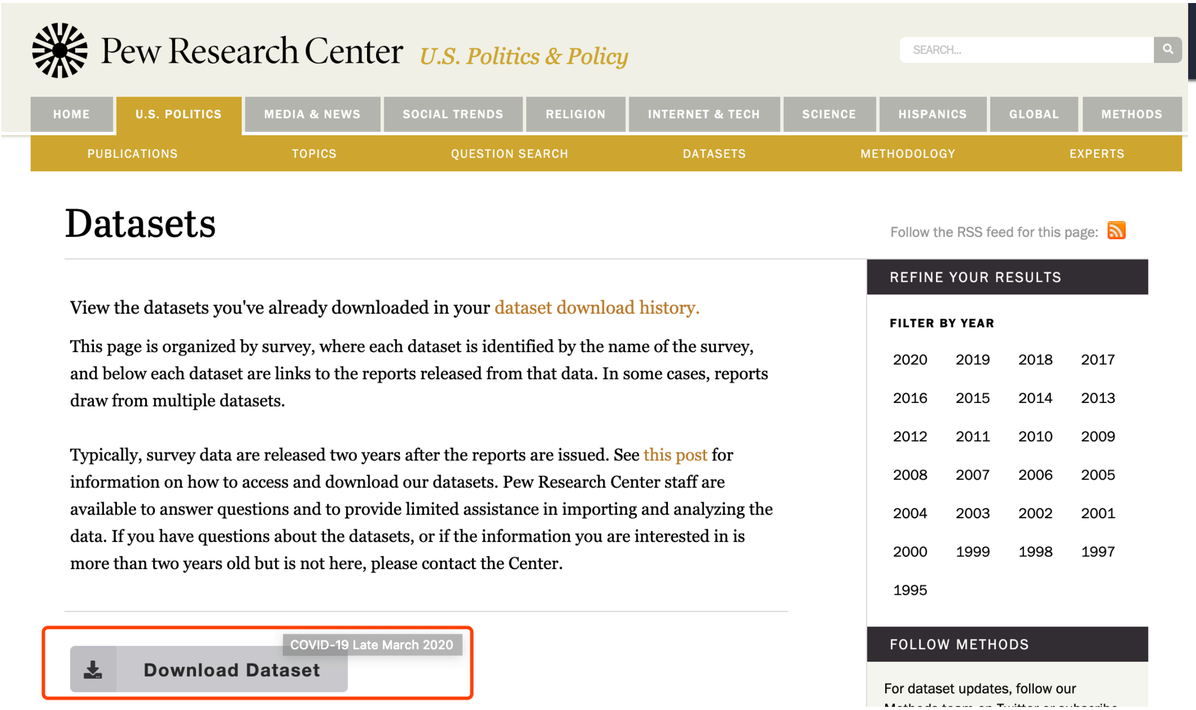
Things + Sense & Communicate

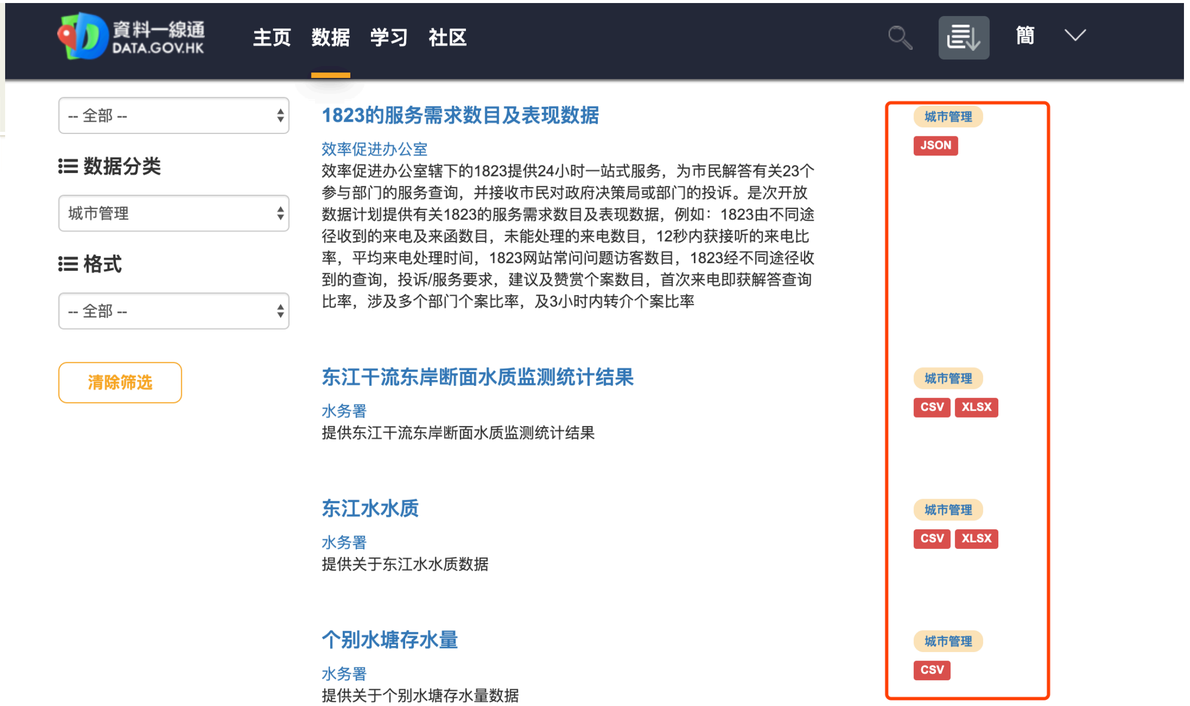
Things can “talk” to other things

Source: Benson Hougland TED Talks

How Can We Get the Data for Further Analysis and Production?

Getting Data - downloading directly





Datasets in different formats:

XLSX, CSV, JSON

Getting Data - scraping

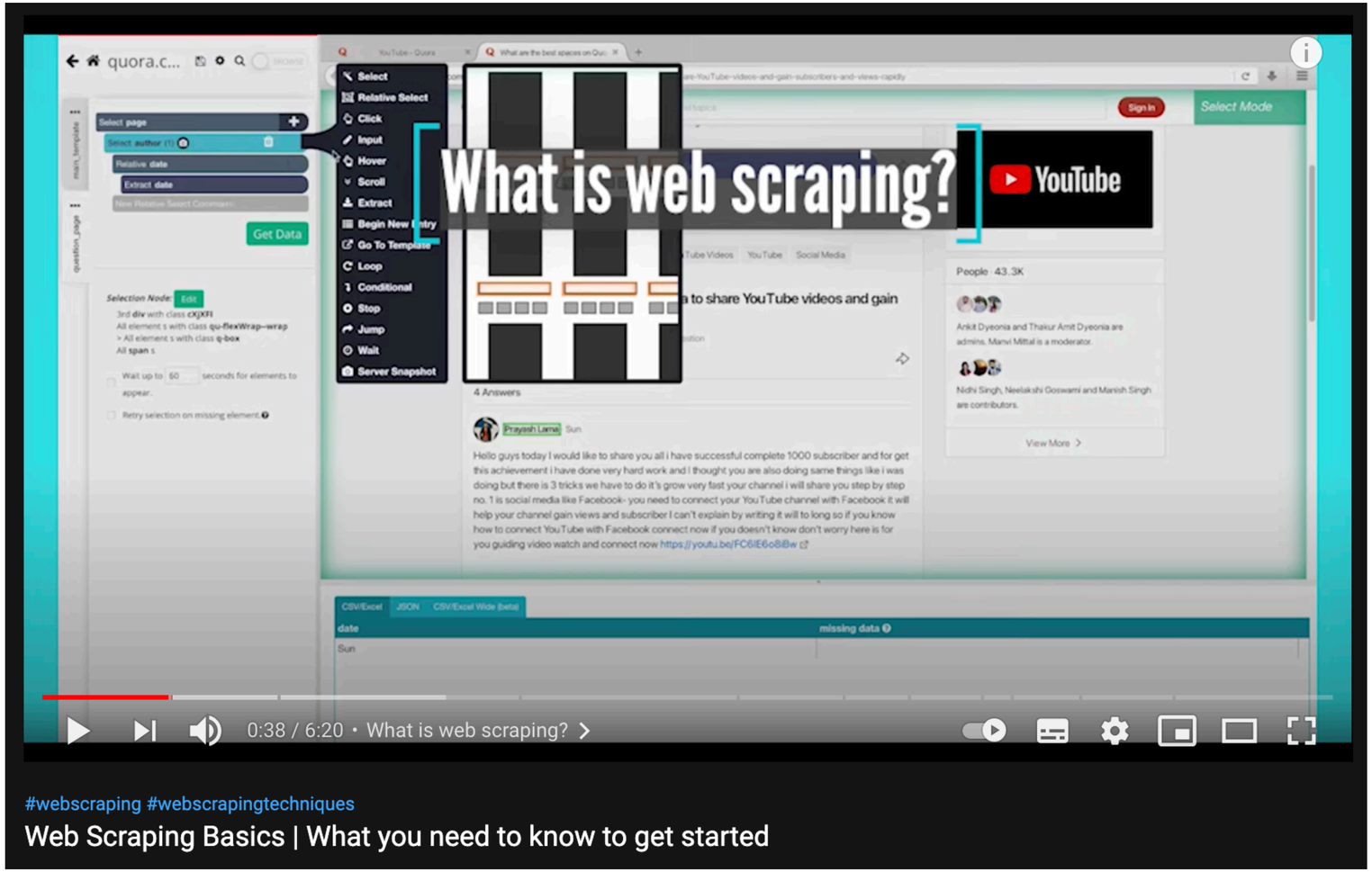
- Scraping website using software:

parsehub

Getting Data - scraping

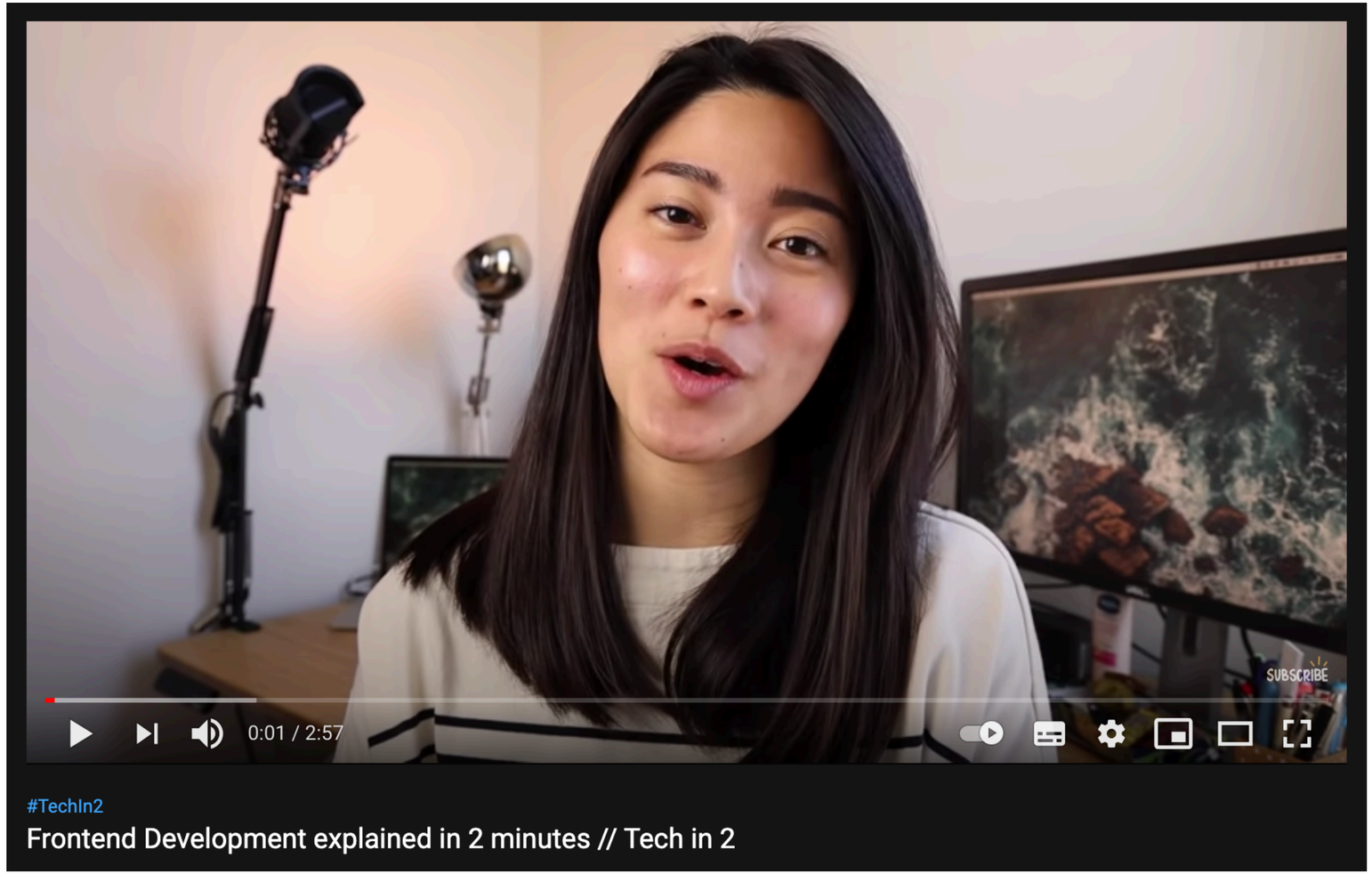
- Website scraping using Python BeautifulSoup

What is web scraping



Source: https://www.youtube.com/watch?v=VyY-w87WHGc

Front-end developing

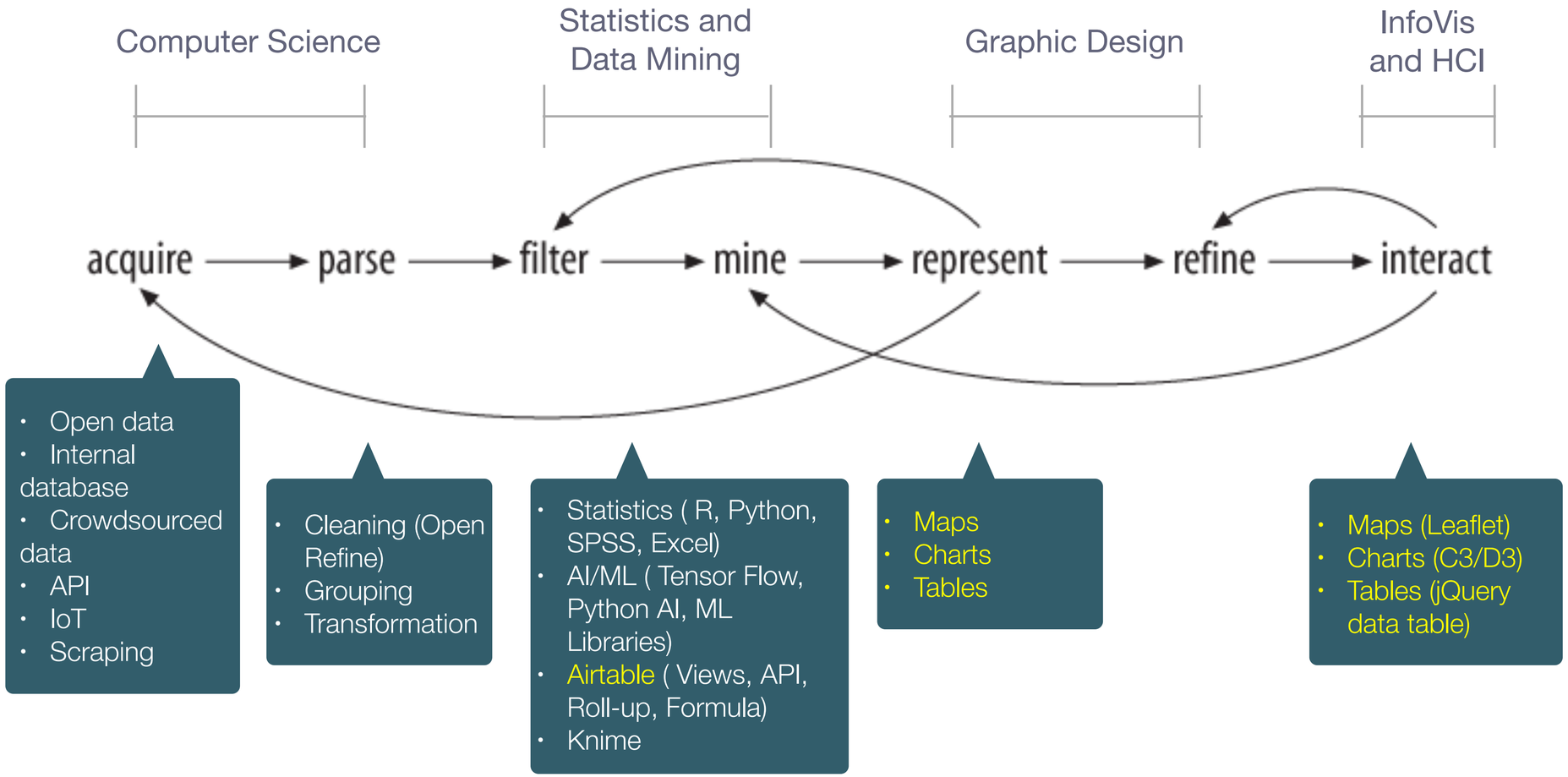


Source: https://www.youtube.com/watch?v=qyHyFsT7Hig

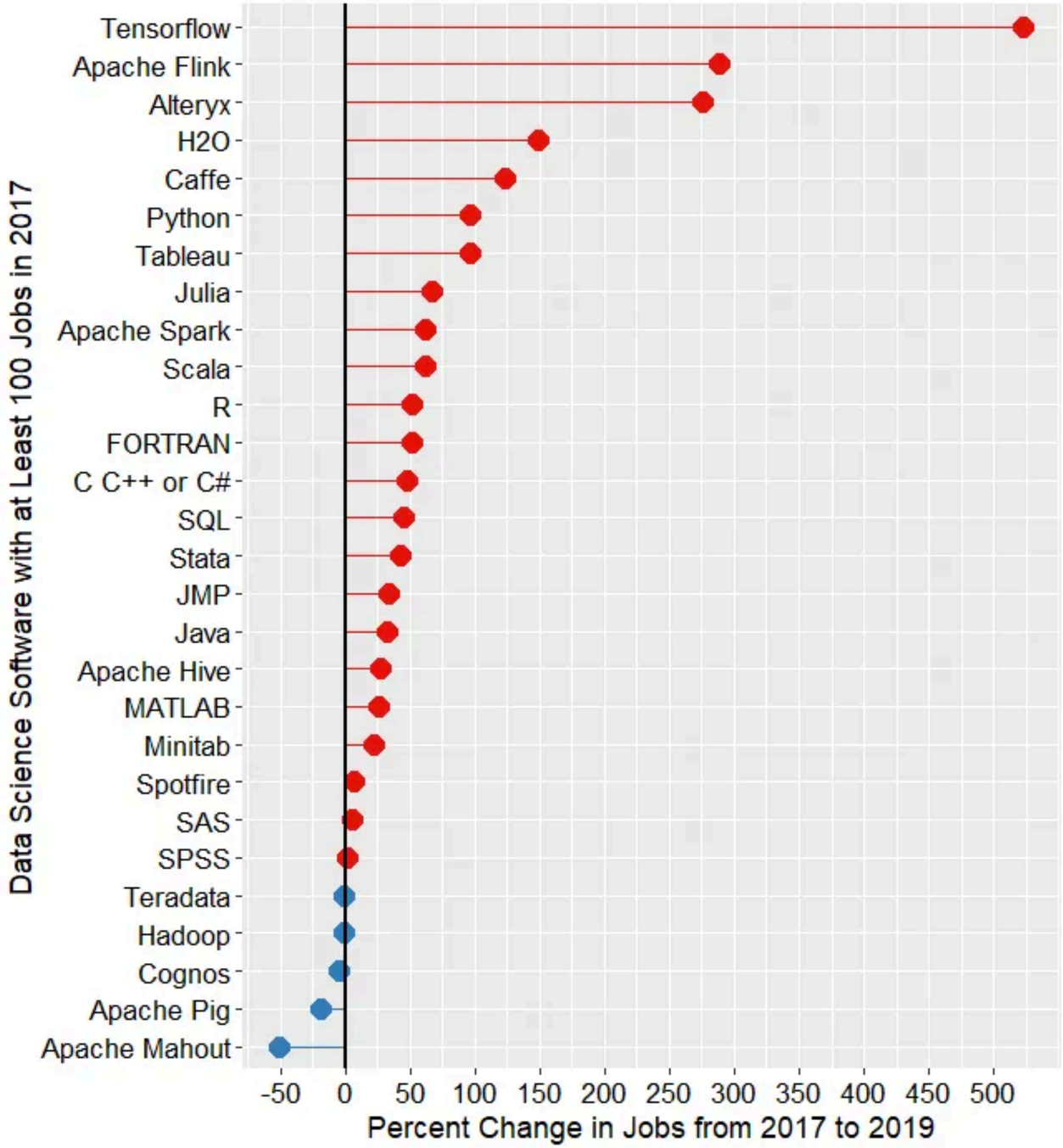
Back-end developing

Source: https://www.youtube.com/watch?v=cbSrsYiRamo

Ben Fry's visualizing data



Source: Ben Fry Visualizing Data



The Popularity of Data Science Software

By Robert A. Muenchen

JavaScript Has Become the Most Popular Front-end Language with Pre-packaged Framework and Library Support

框架	库
e.g. jQuery	e.g. Data table
Angular JS	D3
React	C3D3
Vue	Leaflet

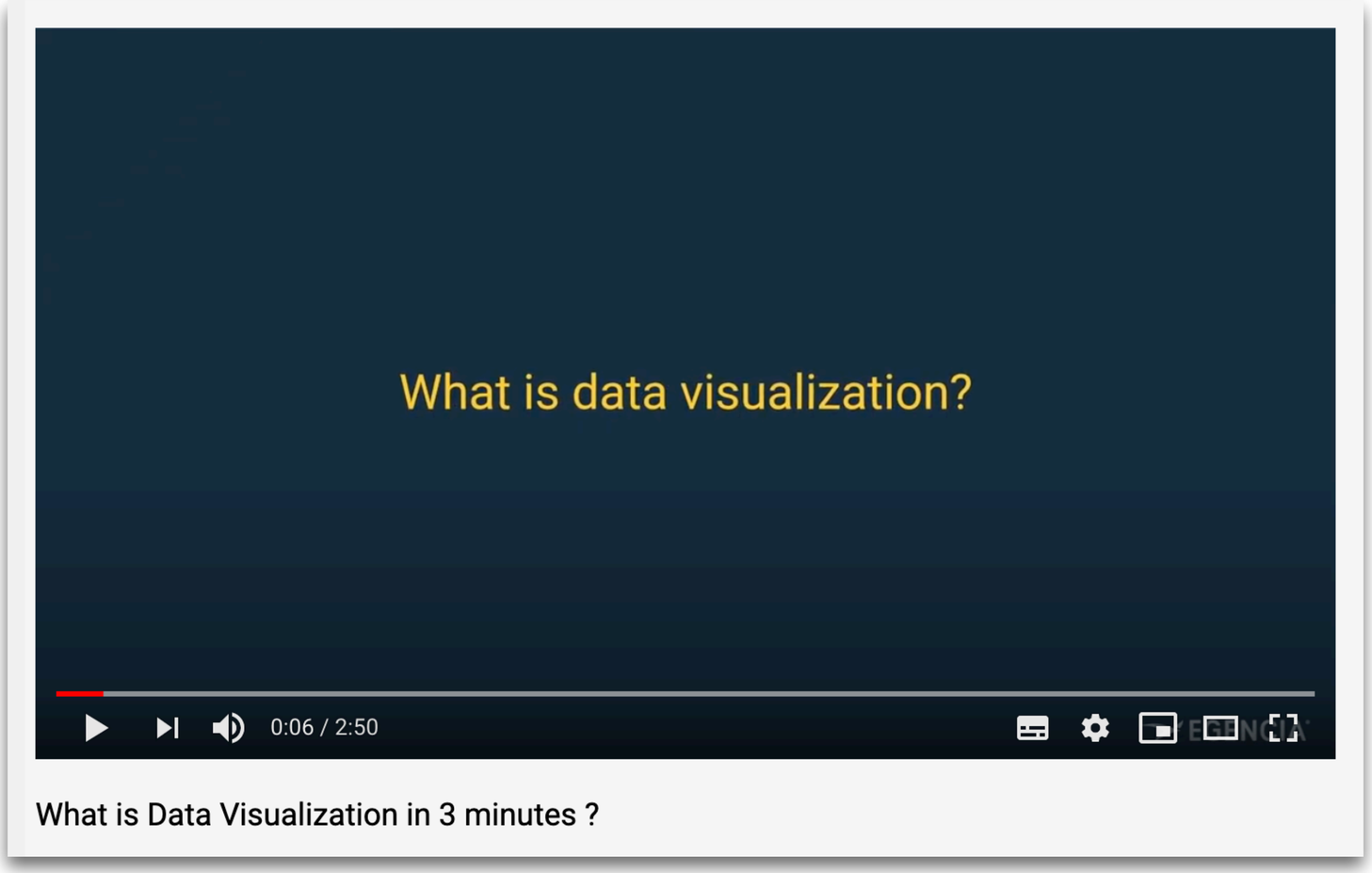
Difference between a framework and a library:

知乎： https://zhuanlan.zhihu.com/p/26078359

Stack Overflow:
https://stackoverflow.com/questions/148747/what-is-the-difference-between-a-framework-and-a-library?noredirect=1&lq=1
https://stackoverflow.com/questions/3057526/framework-vs-toolkit-vs-library

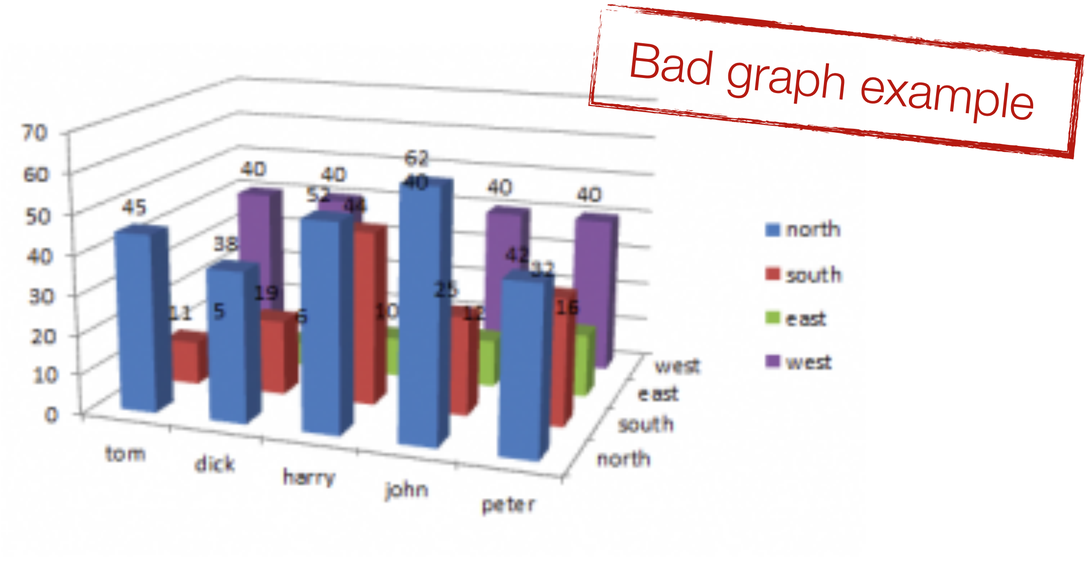
The power of visual communication

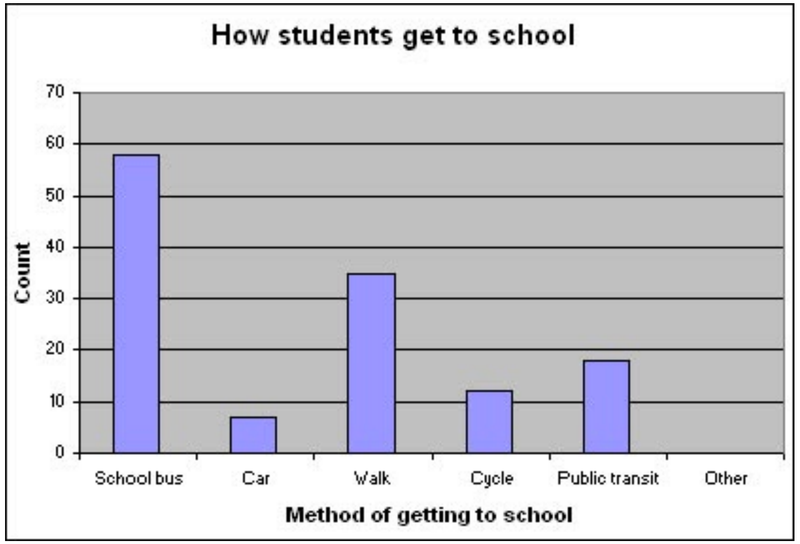
What is data visualization & why is it so important?



YouTube: https://www.youtube.com/watch?v=VyhLRJVoIrI

Basic requirements for graphing





- Scientific title
- Axis labels
- Regular intervals
- Variables on correct axes

- Independent variables(IV) on x axis
- dependent variables(DV) on y axis

- Choosing the right type of graph

Basic requirements for graphing

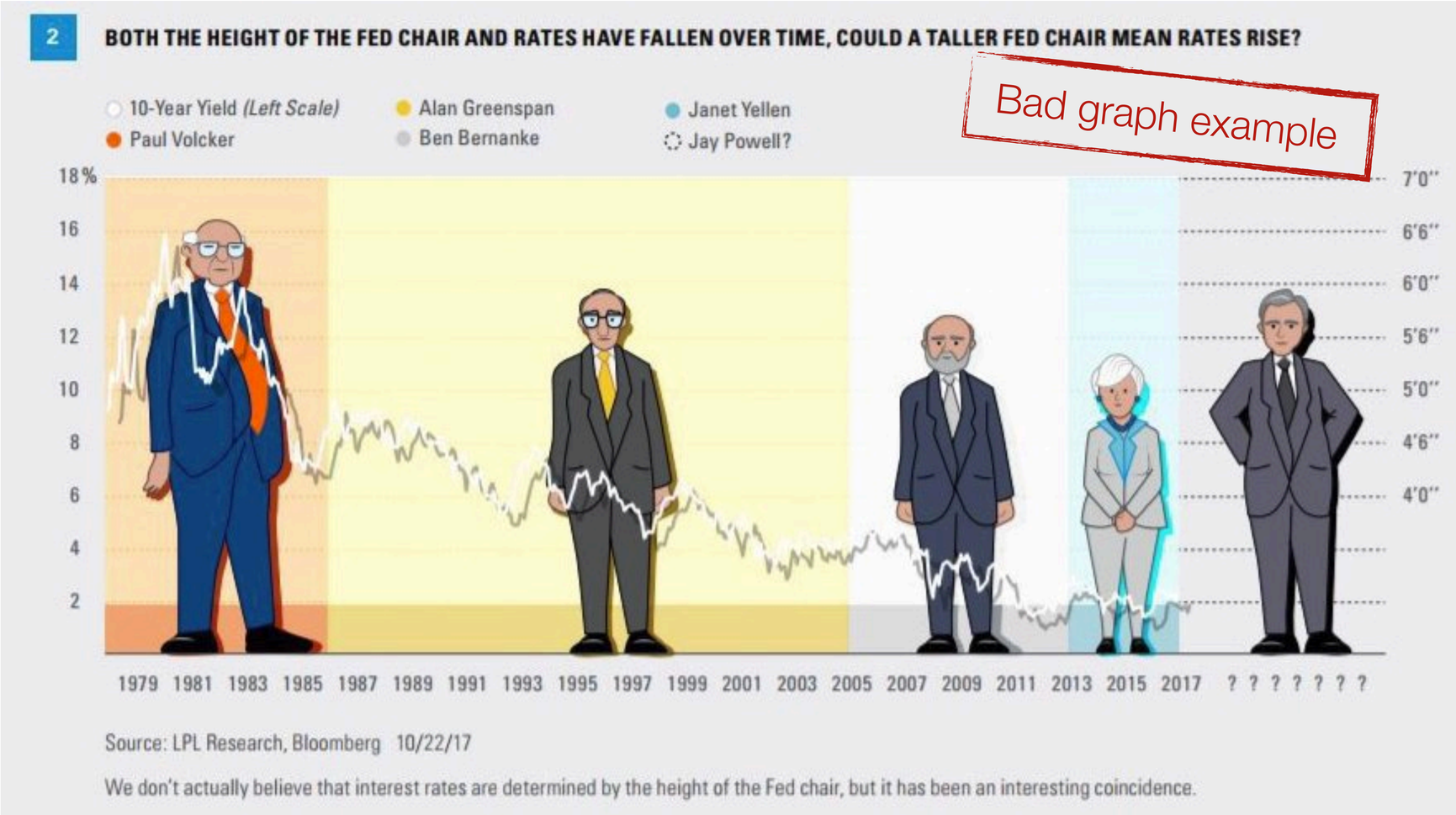


Image source: Twitter@Conner Daley

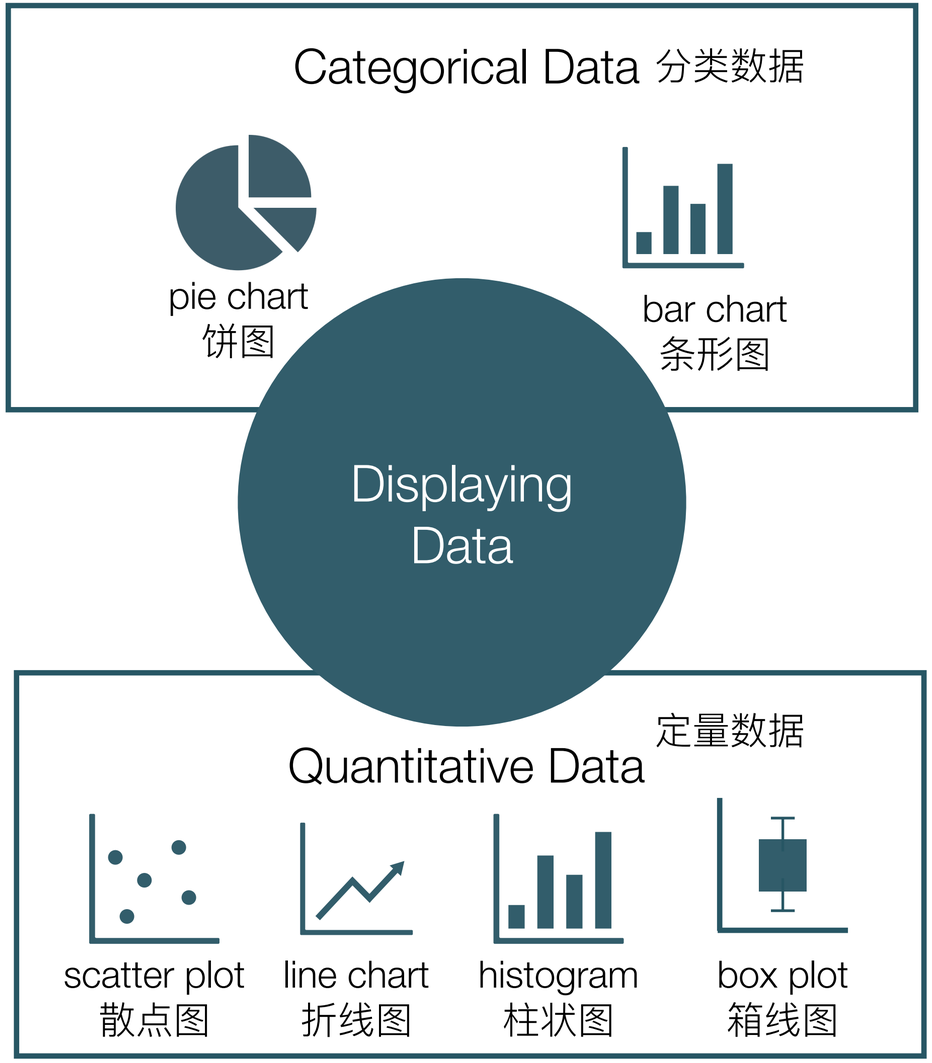
❌ 这张图错误地将美联储主席的身高与利率变化联系起来

Graph Types and Guidelines

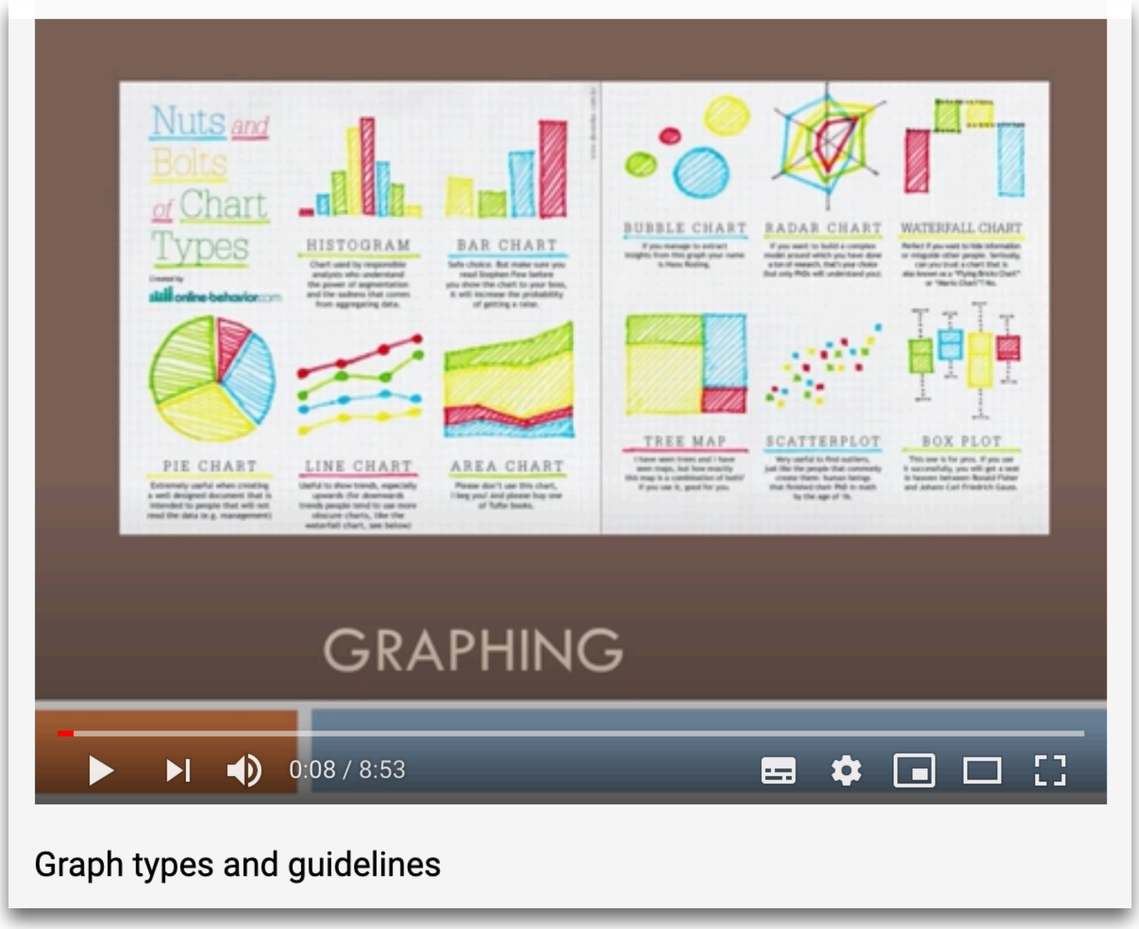
Basic chart types

- Line Chart
- Bar Chart
- Histogram
- Pie Chart
- Scatter Plot
- Box and whisker Plot

Basic chart types



Basic chart types



YouTube: https://www.youtube.com/watch?v=2RCy5hTxuYA

Pie chart 饼图

- A circular statistical graphic, which is divided into slices to illustrate numerical proportion.
- A good use is to show the relationship between parts or percentages of a whole.

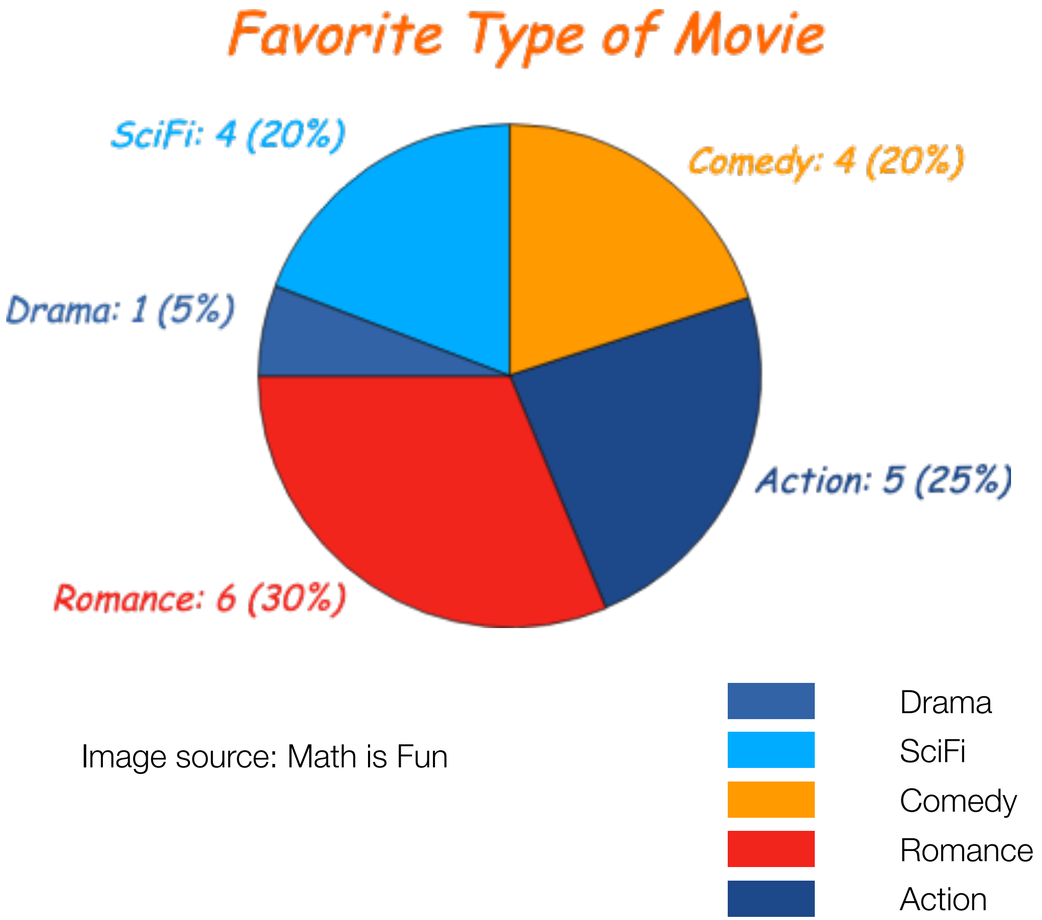


Image source: Math is Fun

Pie chart 饼图

- Why pie charts are bad for business?

🤔🤔🤔

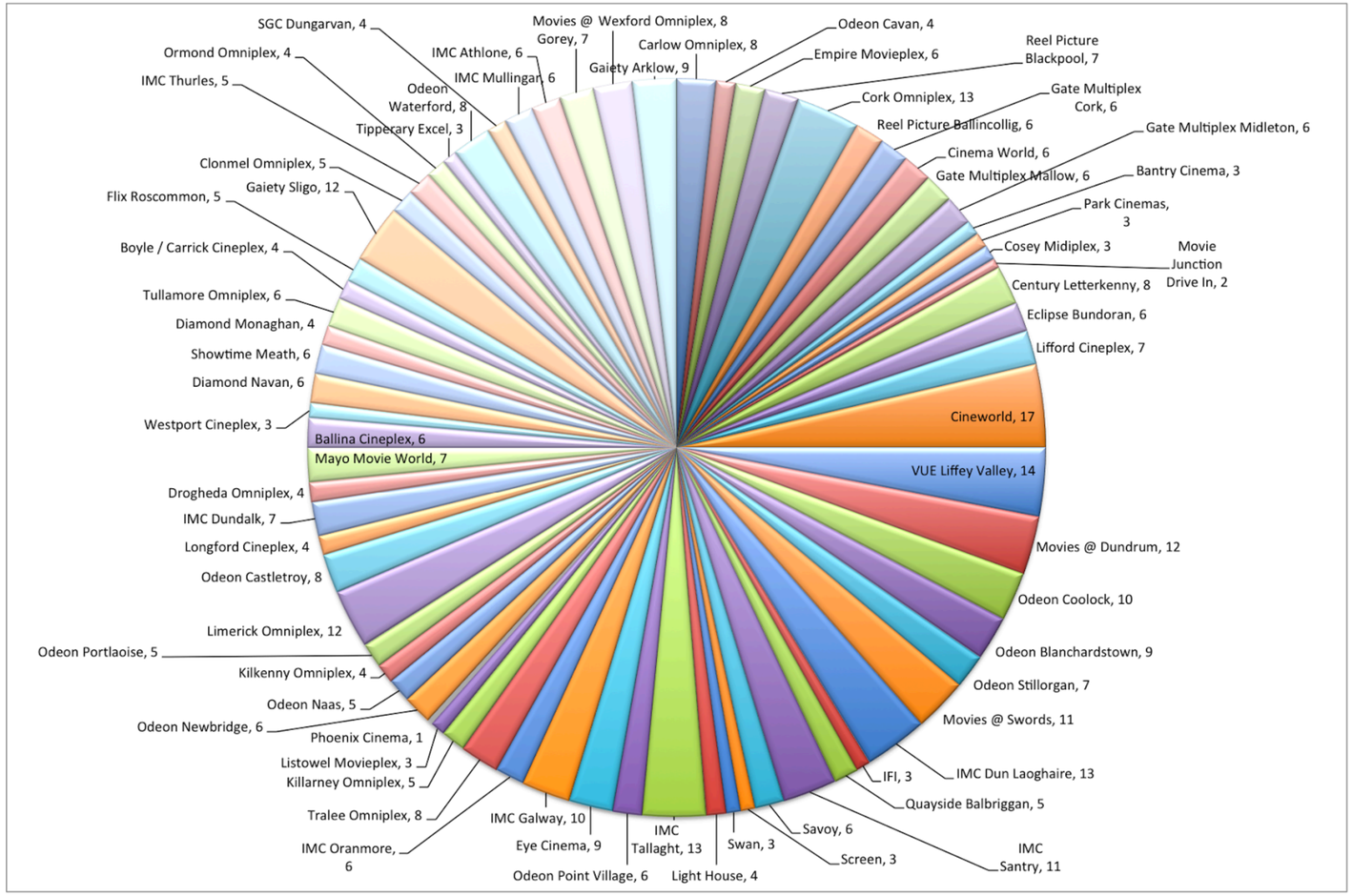


Image source: Method Savvy

Pie chart 饼图

- Why pie charts are bad for business?

  - 数据量多时不易读
  - 不适合做比较（如柱状图更加）
  - 从设计角度考虑，所占空间大

Tips

- 按照比例由大至小或由小至大对各部分排序（方便比较）
- 减少颜色与图标说明
- keep it simple（🙅❌3D）

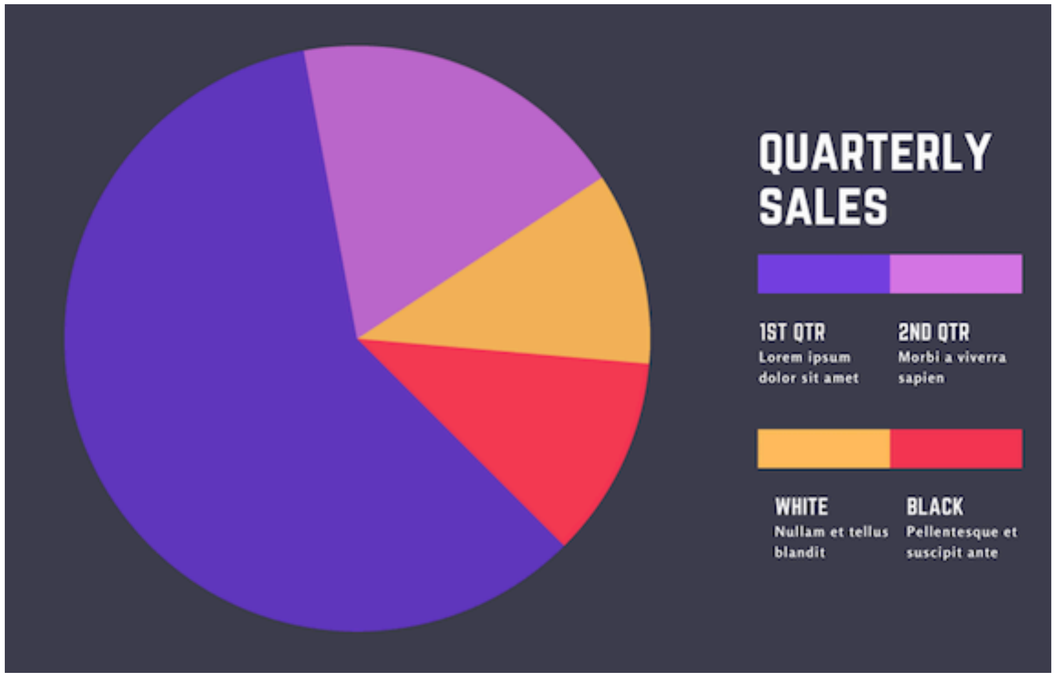


Image source: Canva

Doughnut Chart 环形/圆环图

- Just like a pie chart, a doughnut chart shows the relationship of parts to a whole, but a doughnut chart can contain more than one data series.
- Each data series that you plot in a doughnut chart adds a ring to the chart.

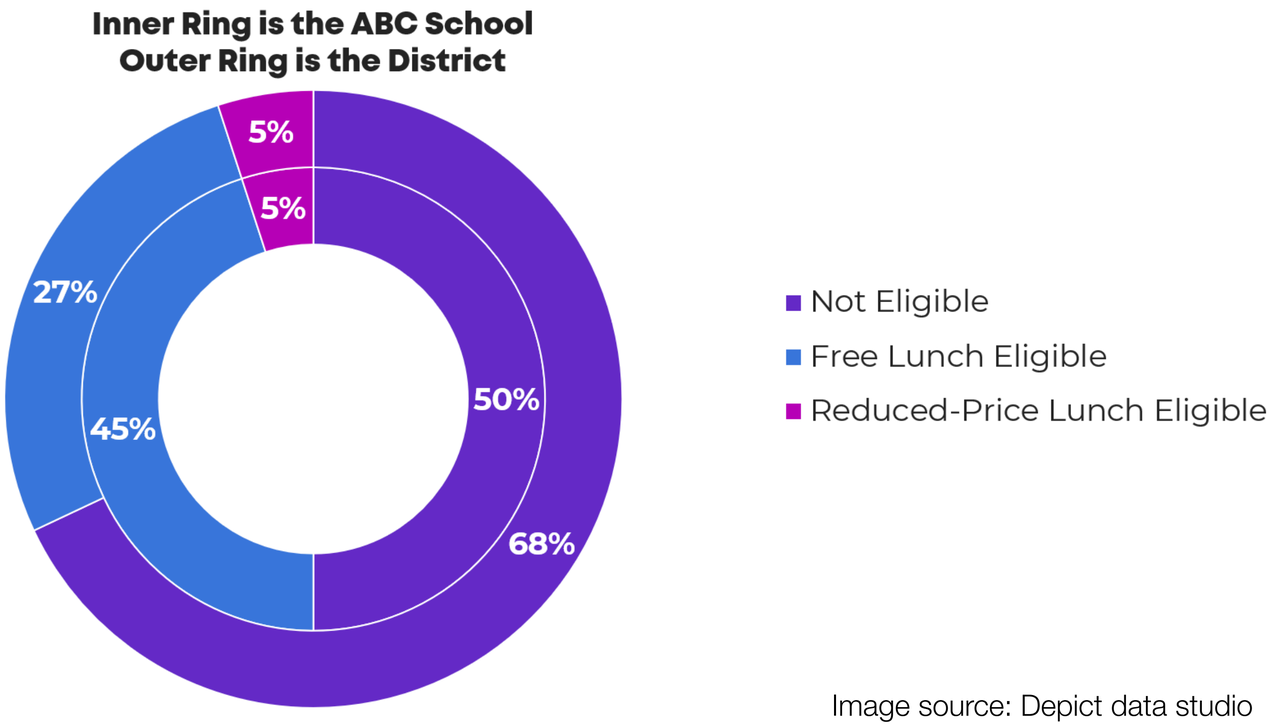
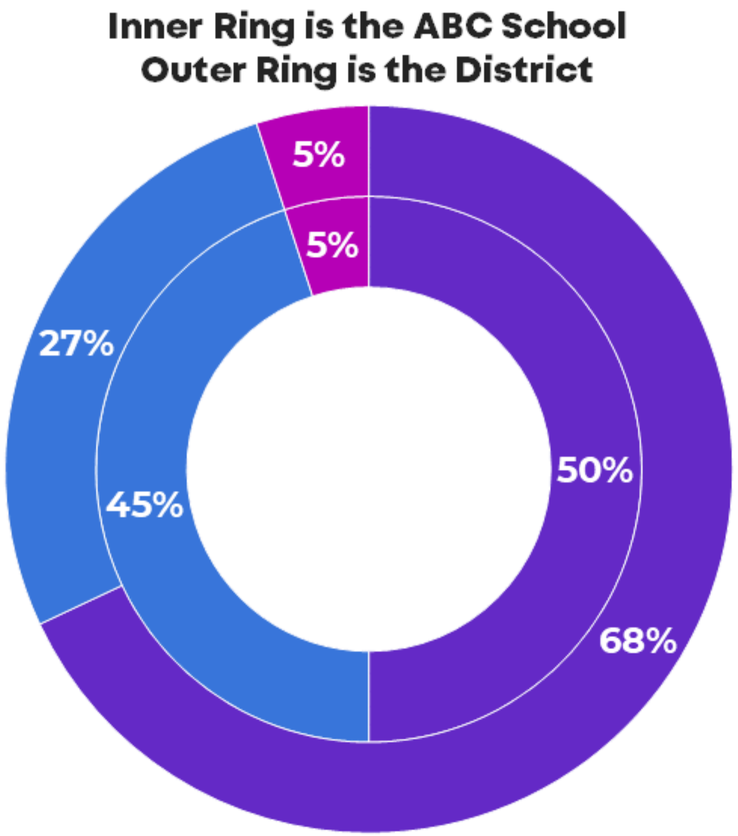


Image source: Depict data studio

Doughnut Chart 环形/圆环图

Free and Reduced Meals



Nested donut chart (Multi-level doughnut chart, Multi-series doughnut chart) allows you to display multiple series in a single visualization. The series are usually related but represent different categories or slices of data (like the distribution of some category values over different years, etc.)

- Not Eligible
- Free Lunch Eligible
- Reduced-Price Lunch Eligible

depict data studio

Pie-in-a-Donut chart

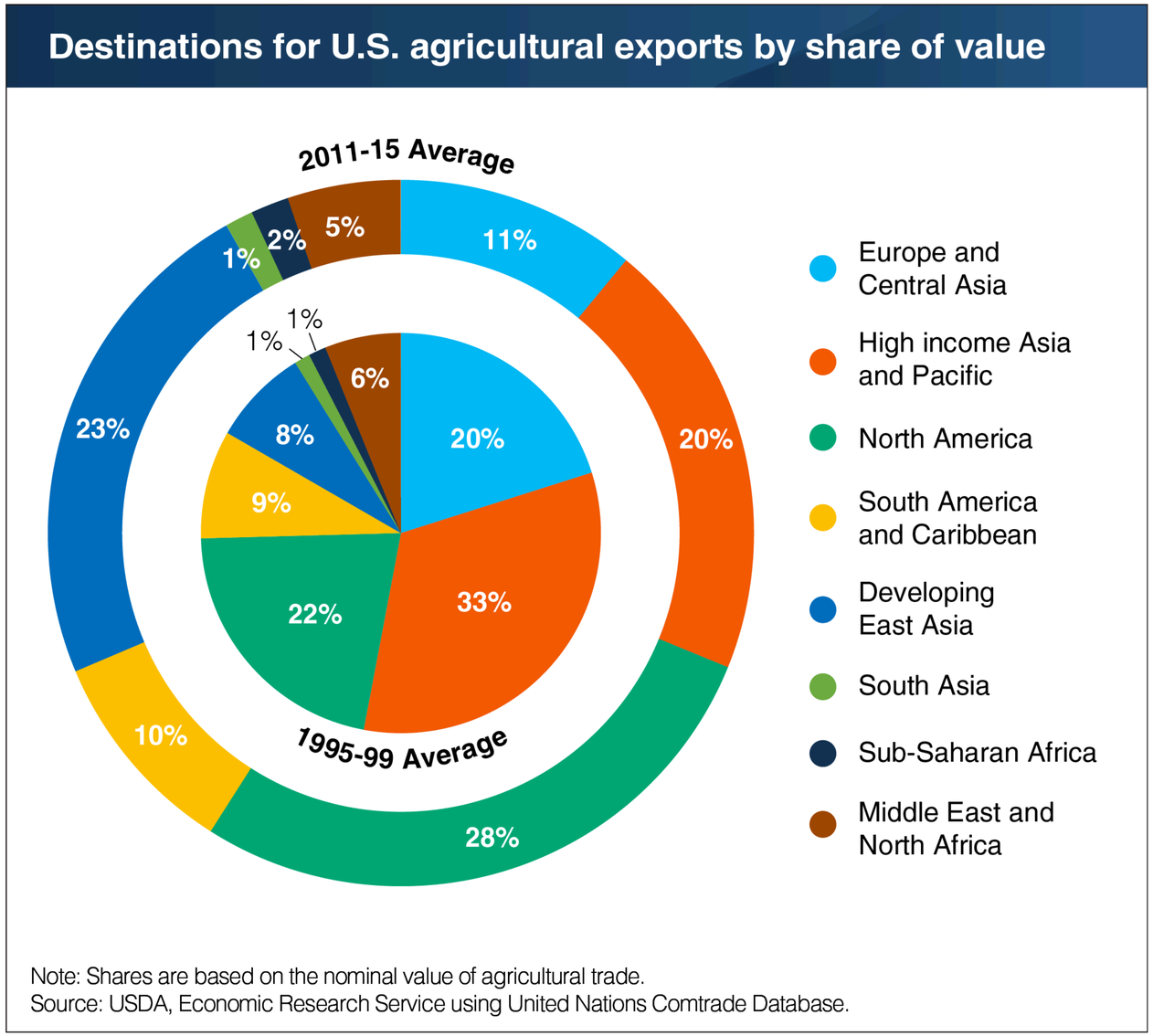
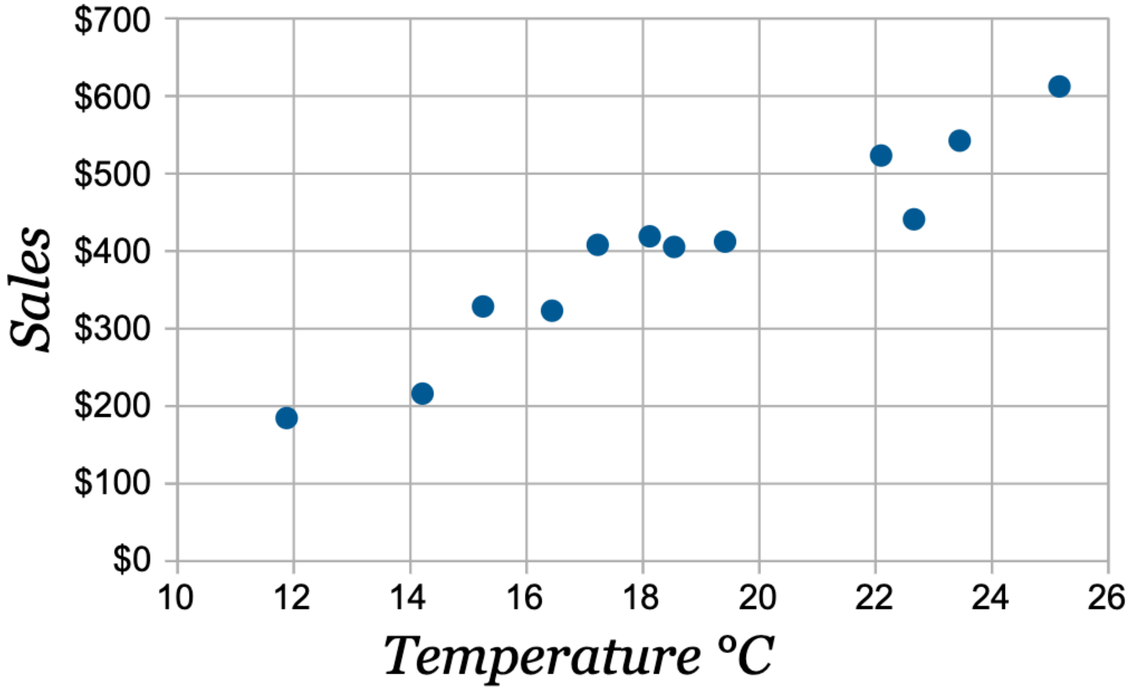


Image source: PolicyViz

Scatter plot 散点图

- A Scatter (XY) Plot has points that show the relationship between two sets of data.



Ice Cream Sales vs Temperature	
Temperature °C	**Ice Cream Sales**
14.2°	$215
16.4°	$325
11.9°	$185
15.2°	$332
18.5°	$406
22.1°	$522
19.4°	$412
25.1°	$614
23.4°	$544
18.1°	$421
22.6°	$445
17.2°	$408

Image source: Math is Fun

Scatter plot 散点图

- It is now easy to see that warmer weather leads to more sales.
- We can also draw a "**Line of Best Fit**"(also called a "Trend Line")最适配合线 on our scatter plot:

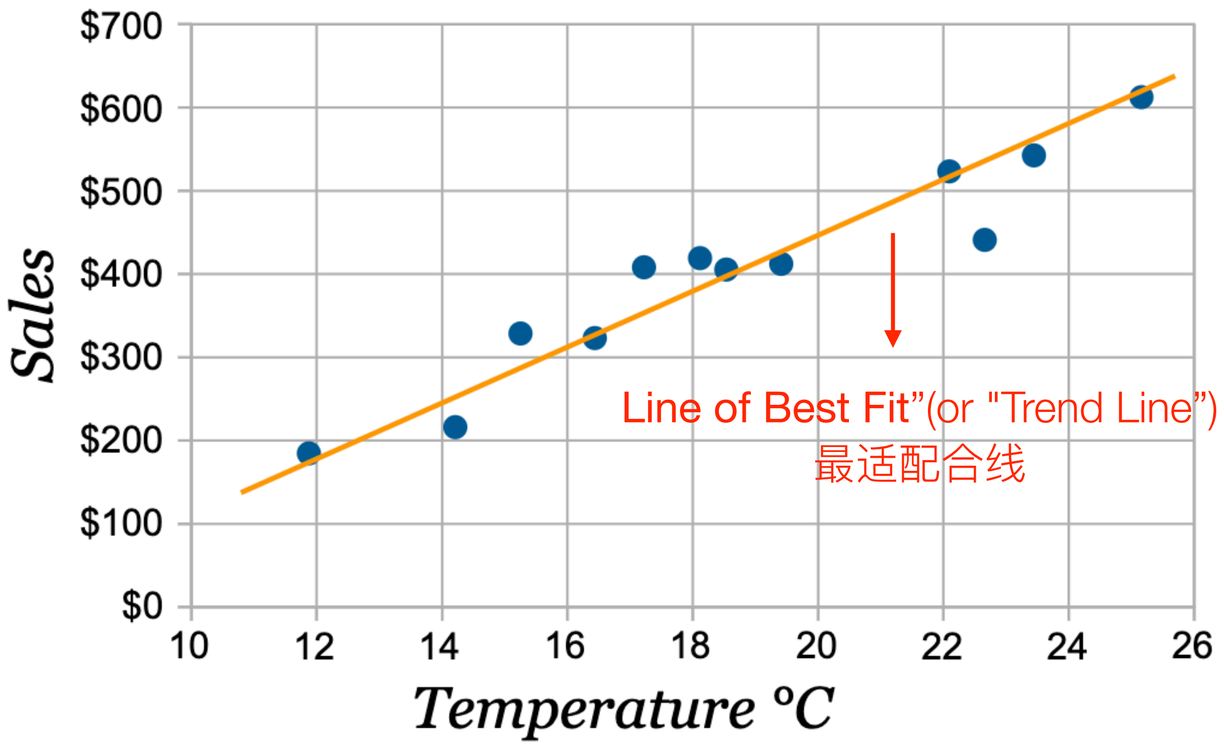
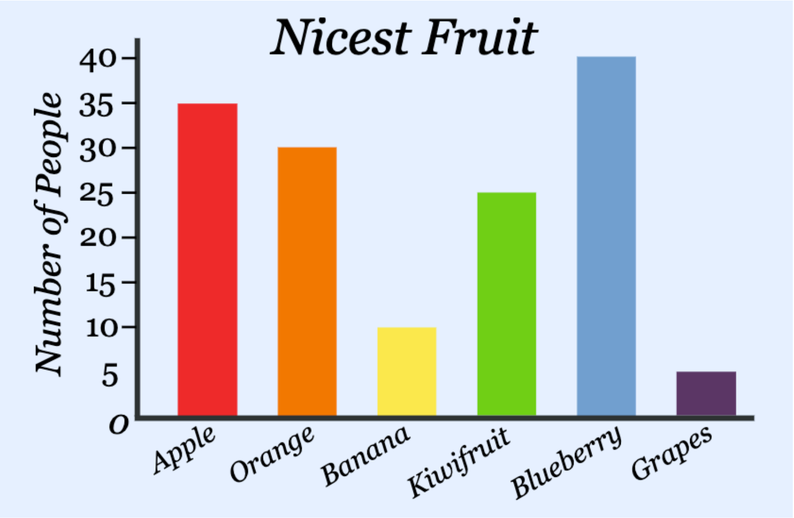


Image source: Math is Fun

Bar Chart 条形图

- A bar chart or bar graph is a chart or graph that presents **categorical data** with rectangular bars with heights or lengths proportional to the values that they represent.
  —Wikipedia

Example: A survey of 145 people asked them "Which is the nicest fruit?”



Bar Graphs can also be Horizontal:

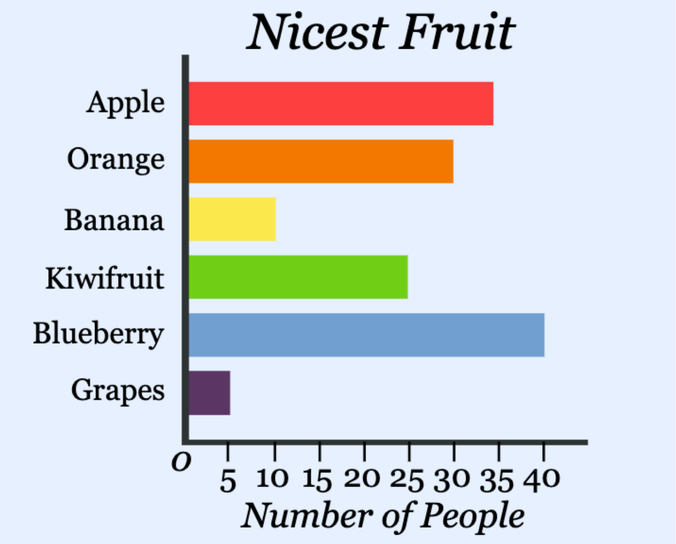
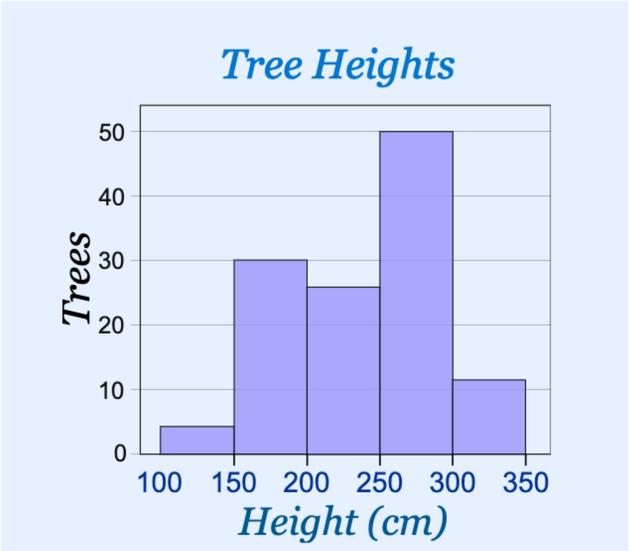


Image source: Math is Fun

Histgram 柱状图/直方图

- A histogram is an approximate representation of the distribution of **numerical or categorical data**. It was first introduced by Karl Pearson. —Wikipedia
- It is similar to a Bar Chart, but a histogram groups numbers into **ranges**.
- Histograms are a great way to show results of **continuous data(连续数据),** such as: weight, height, time, etc.



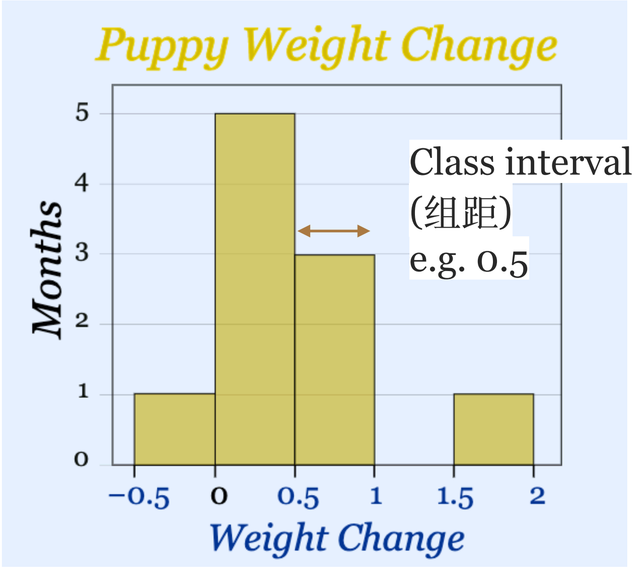


Image source: Math is Fun

Histgram 柱状图/直方图

- **Frequency Histogram (频率直方图)**

  A Frequency Histogram is a special graph that uses vertical columns to show frequencies (how many times each score occurs).

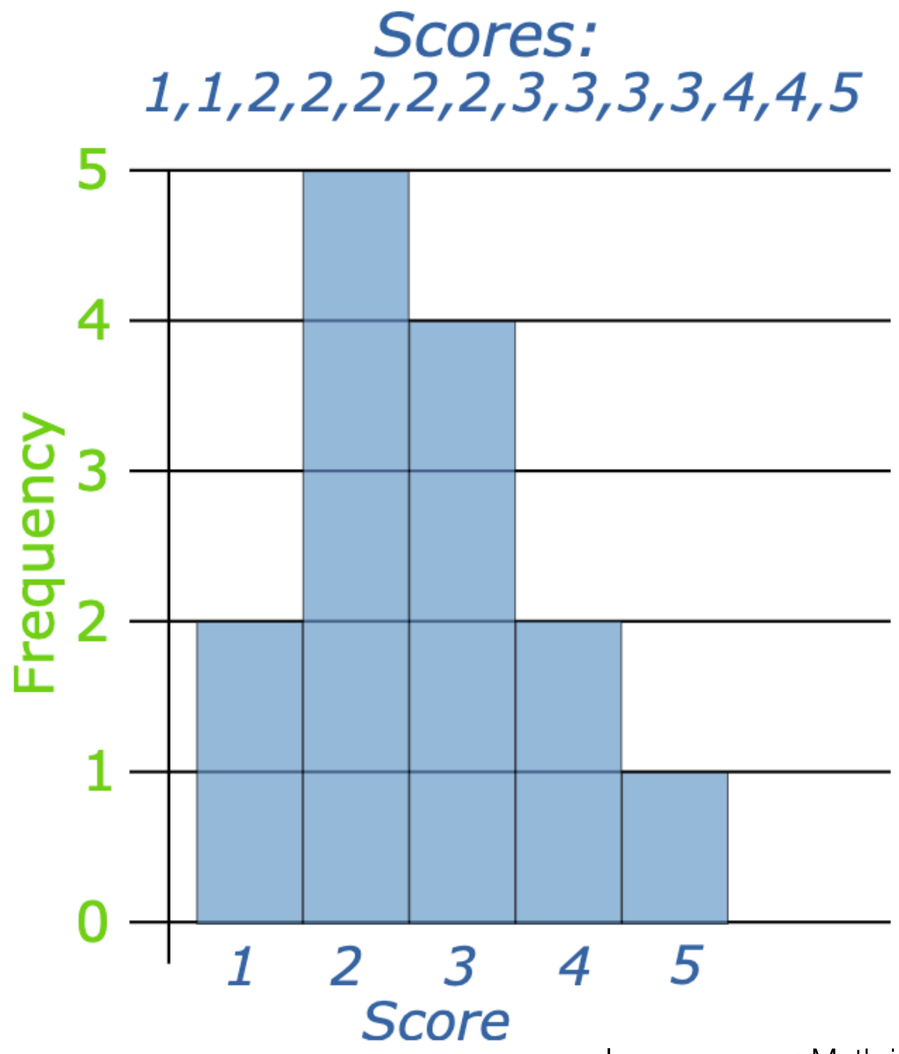


Image source: Math is Fun

Histogram have wide applications in statistics

A histogram(直方图) is NOT a bar chart(条形图)

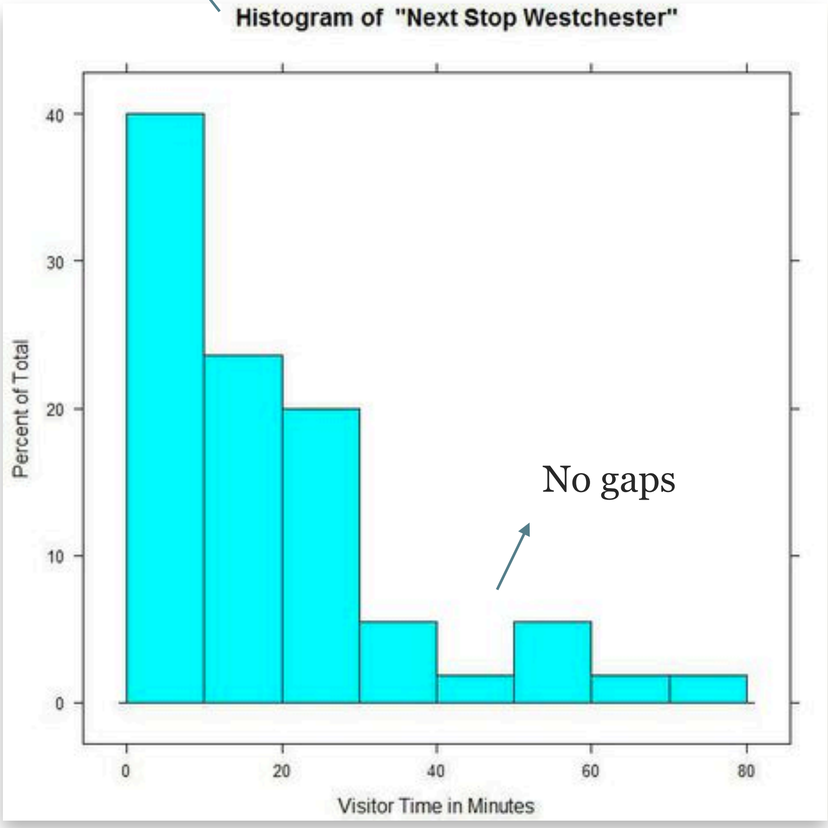


Figure 1. A histogram showing the distribution of times visitors spent at one museum exhibition.

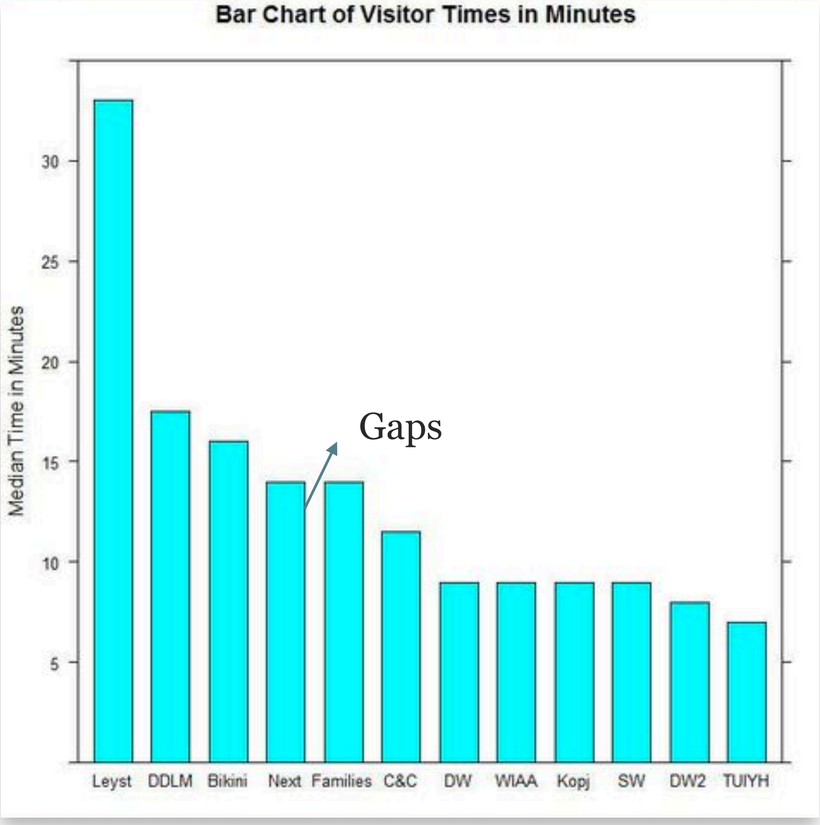


Figure 2. A bar chart comparing the median times visitors stayed at each of 12 museum exhibitions.

- Histograms are:

- used to show distributions of variables.
- used for data that is at least at the <u>ordinal level (定序变量)</u> of measurement.

- The bars can not be rearranged.

- Bar charts are:

- used to compare variables.
- used for data at the <u>nominal level (定类变量)</u> of measurement.

- The bars can be rearranged to decrease height.

Box and whisker plot 箱形图

- In descriptive statistics, a box plot or box plot is a method for graphically depicting groups of numerical data through their quartiles (四分位数).
  —Wikipedia

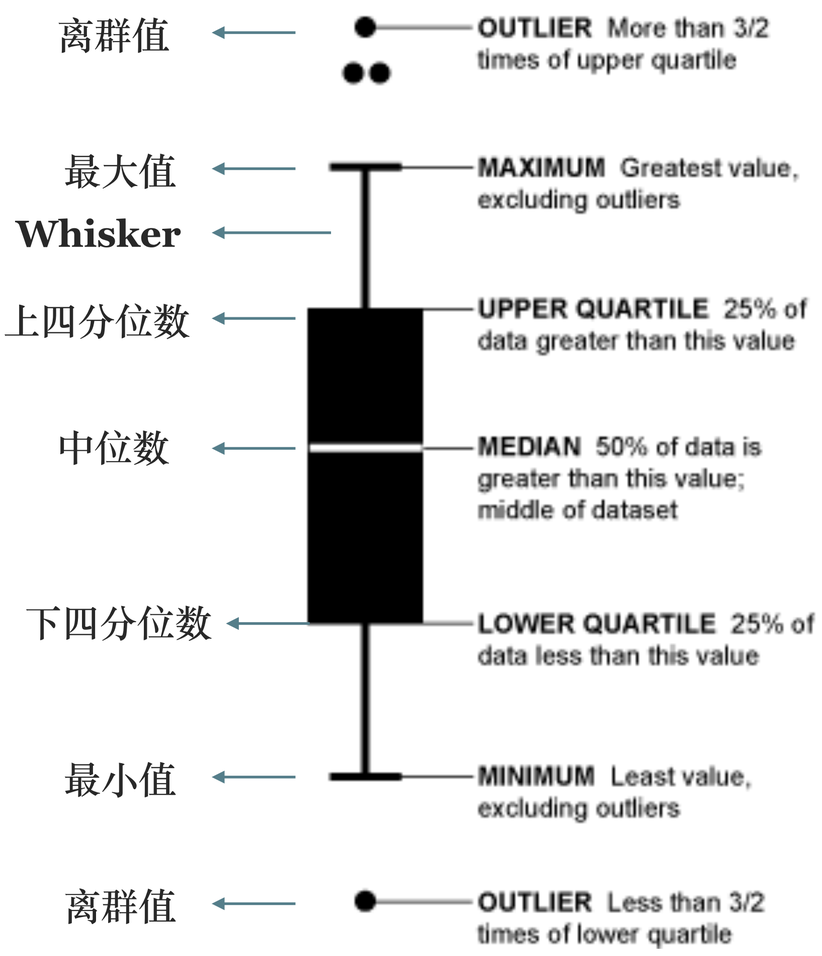
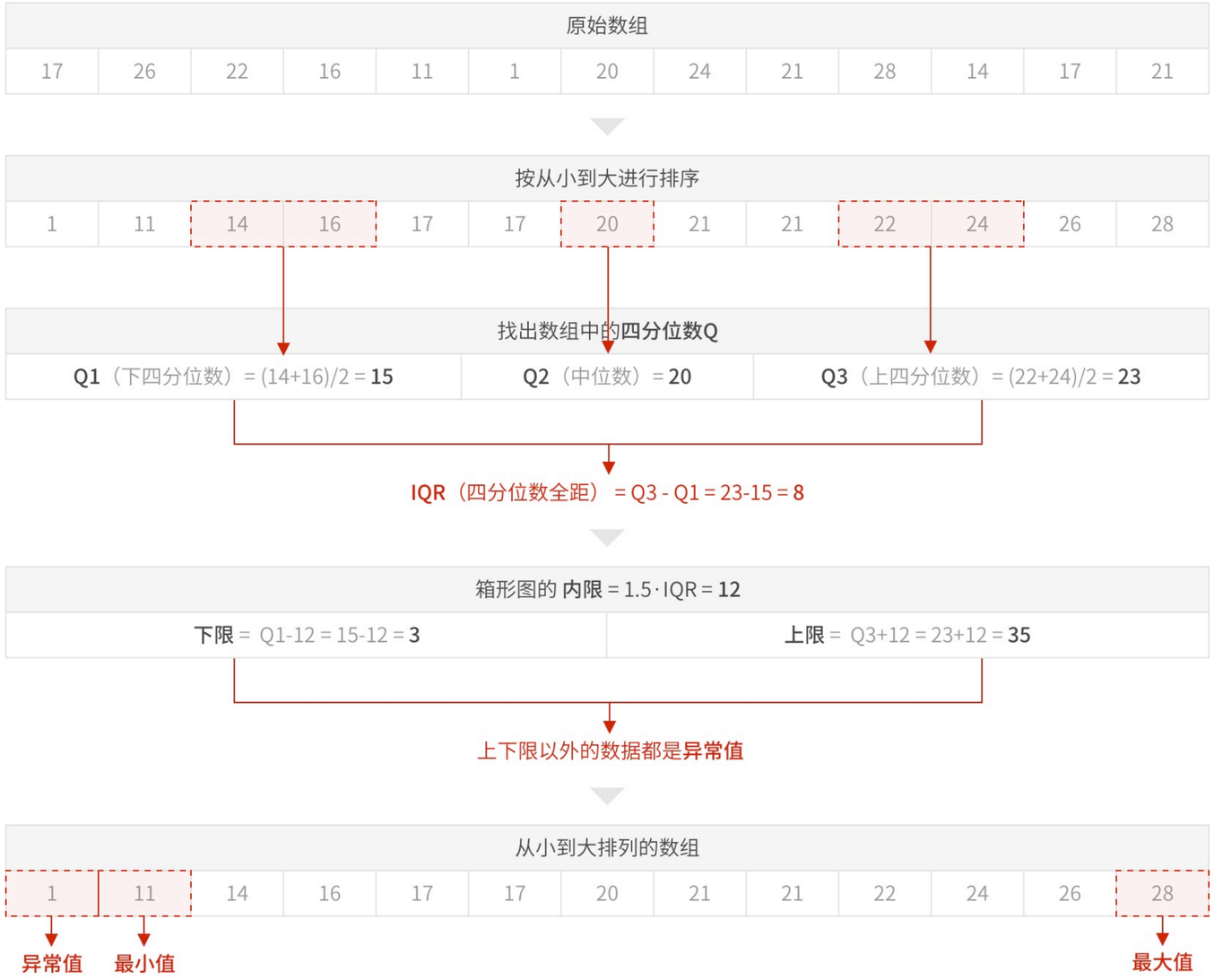


Image source: FlowingData

Box and whisker plot 箱形图



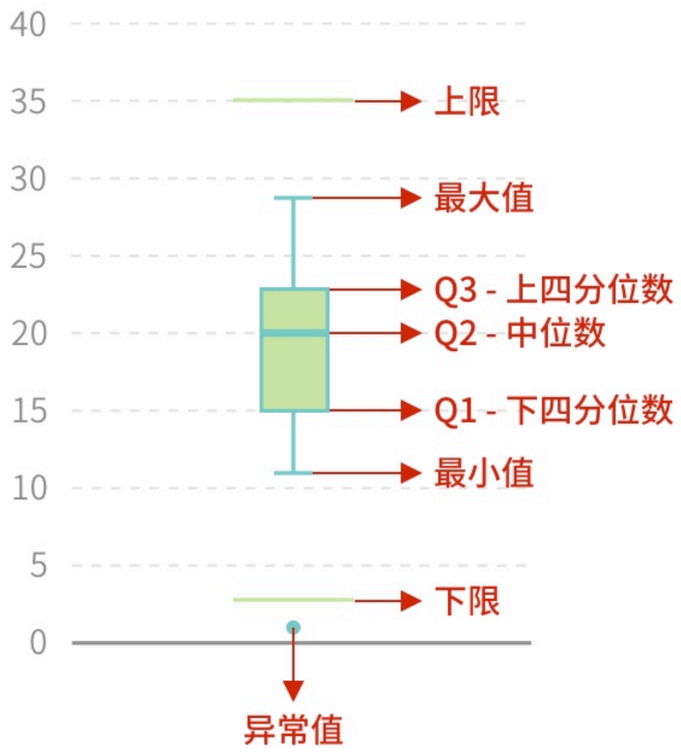


Image source: tuzhidian

Box and whisker plot 箱形图

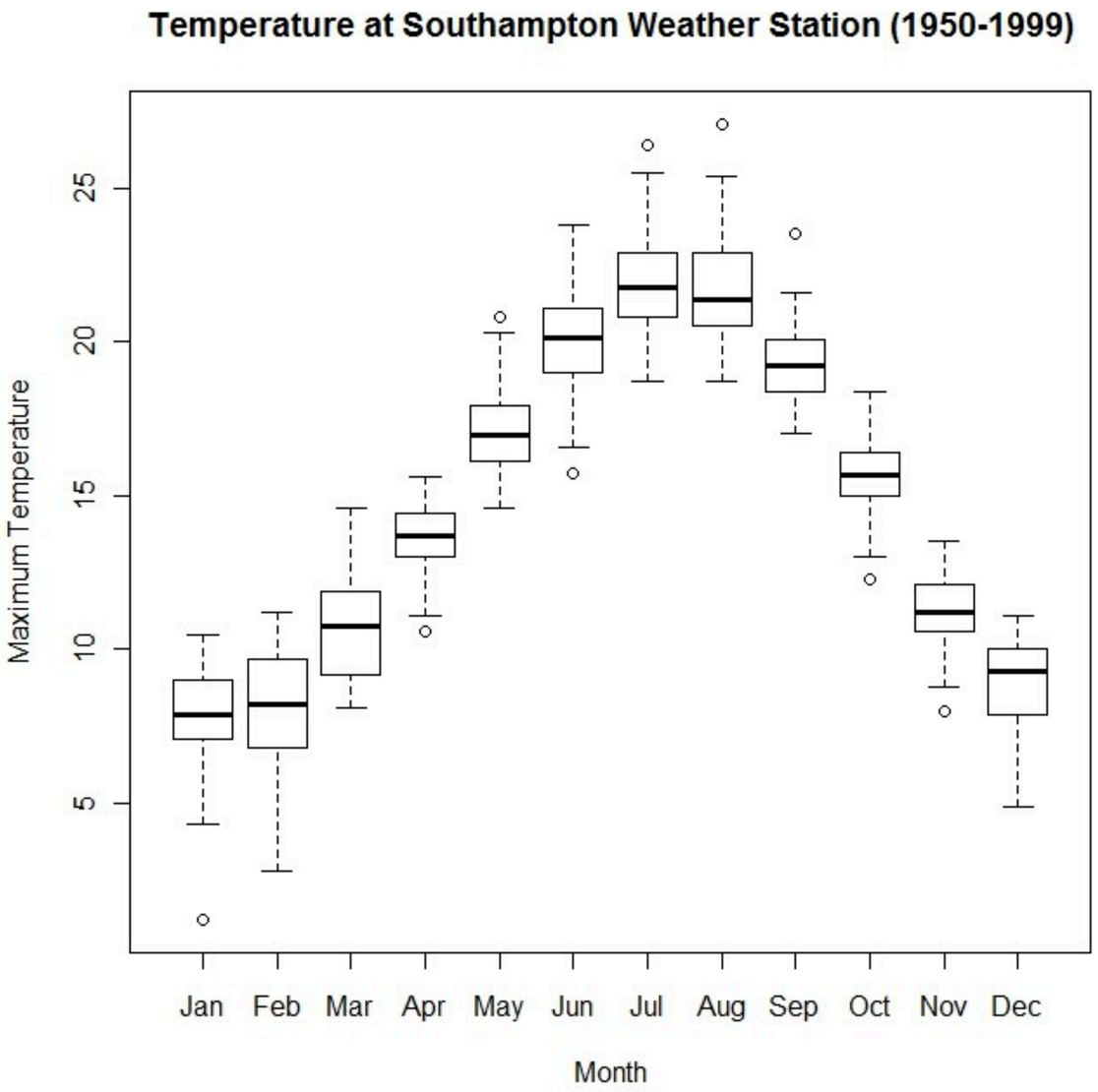


Image source: R-bloggers

Line chart 折线图

- A line chart or line plot or line graph or curve chart is a type of chart which displays information as a series of data points called 'markers' connected by straight line segments.
- It is a basic type of chart common in many fields.

—Wikipedia

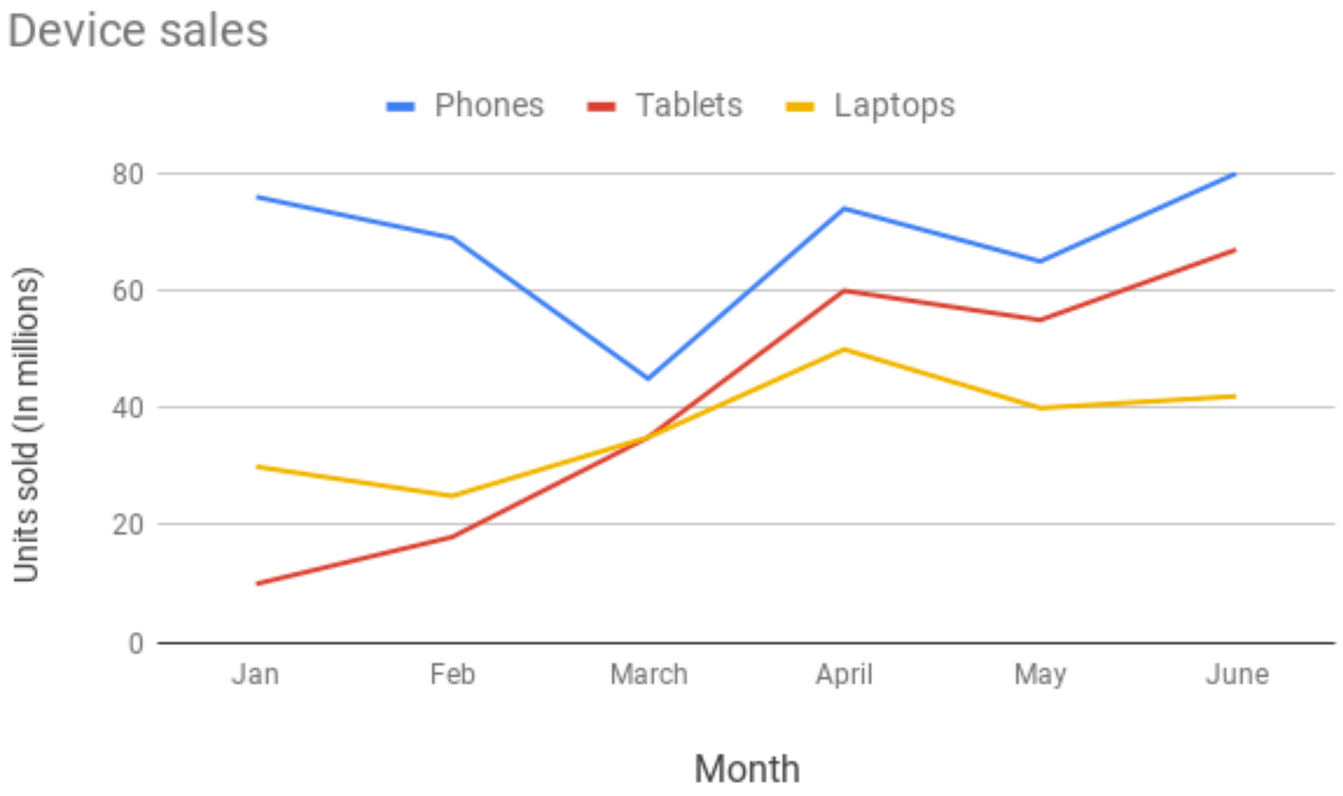


Image source: Google Support

How to choose the right type of graph?

Chart type	Best used to
Pie chart 饼图	show and compare part or percentage of the data to the whole 展示与比较整体数据中各部分或其百分比（的大小）
Bar chart 条形图	show and compare the number for different categories of data 展示与比较不同类别数据的数值大小
Histgram 直方图/柱状图	represent of the distribution of numerical or categorical data 展示数值数据或分类数据的分布情况
Line chart 折线图	show change over a period of time 展示一段时间的变化
Scatter plot 散点图	show how much one variable is affected by another 展示一个变量被另一个变量的影响程度
Box and whiskers plot 箱形图	show measures of variation 衡量变量间的变动量

Ask yourself 3 questions before encoding data

What graphic forms are more appropriate to represent your data?

1. Think about the audience and the publication
2. Think of the questions your graphic should help readers answer
3. Can you understand it without reading every single figure

Source: Alberto Cairo, University of Miami

5 Main Usages for Distinction of Chart Types

- Presenting distribution
- Visualizing the compositions
- Showing the relationships
- Indicating the trend
- Comparing

Source: Lynia Li - medium

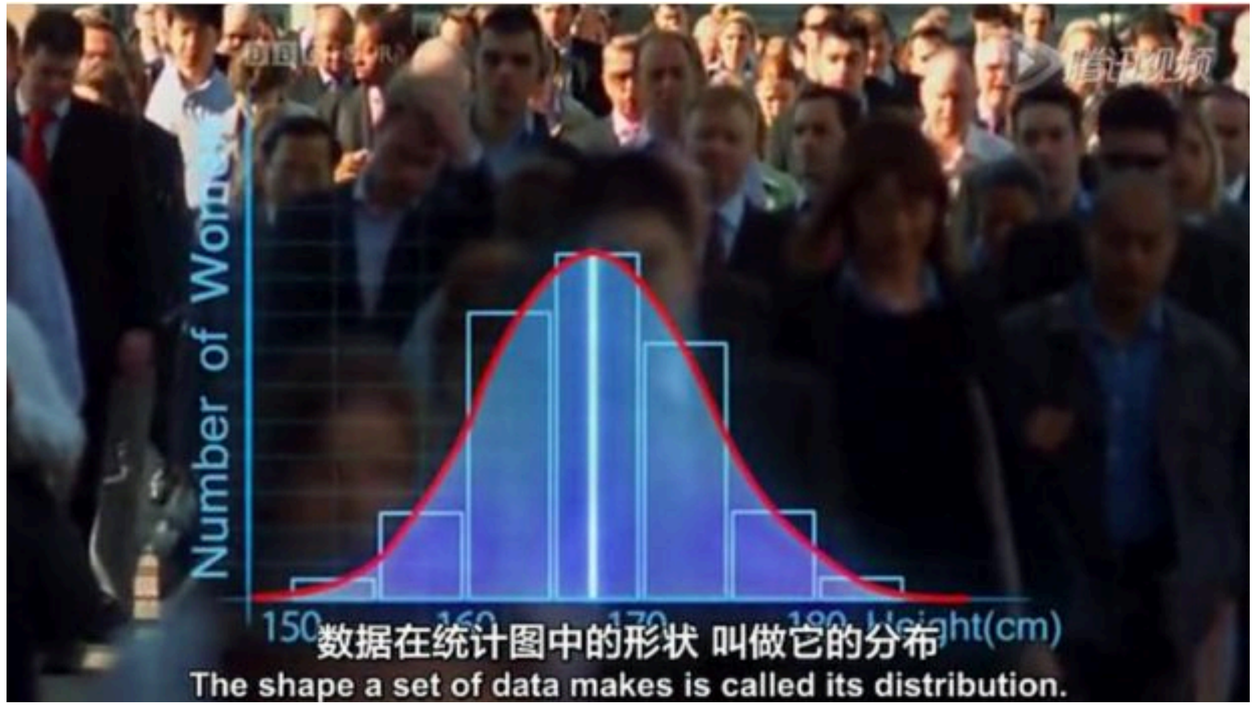


Image credit: BBC

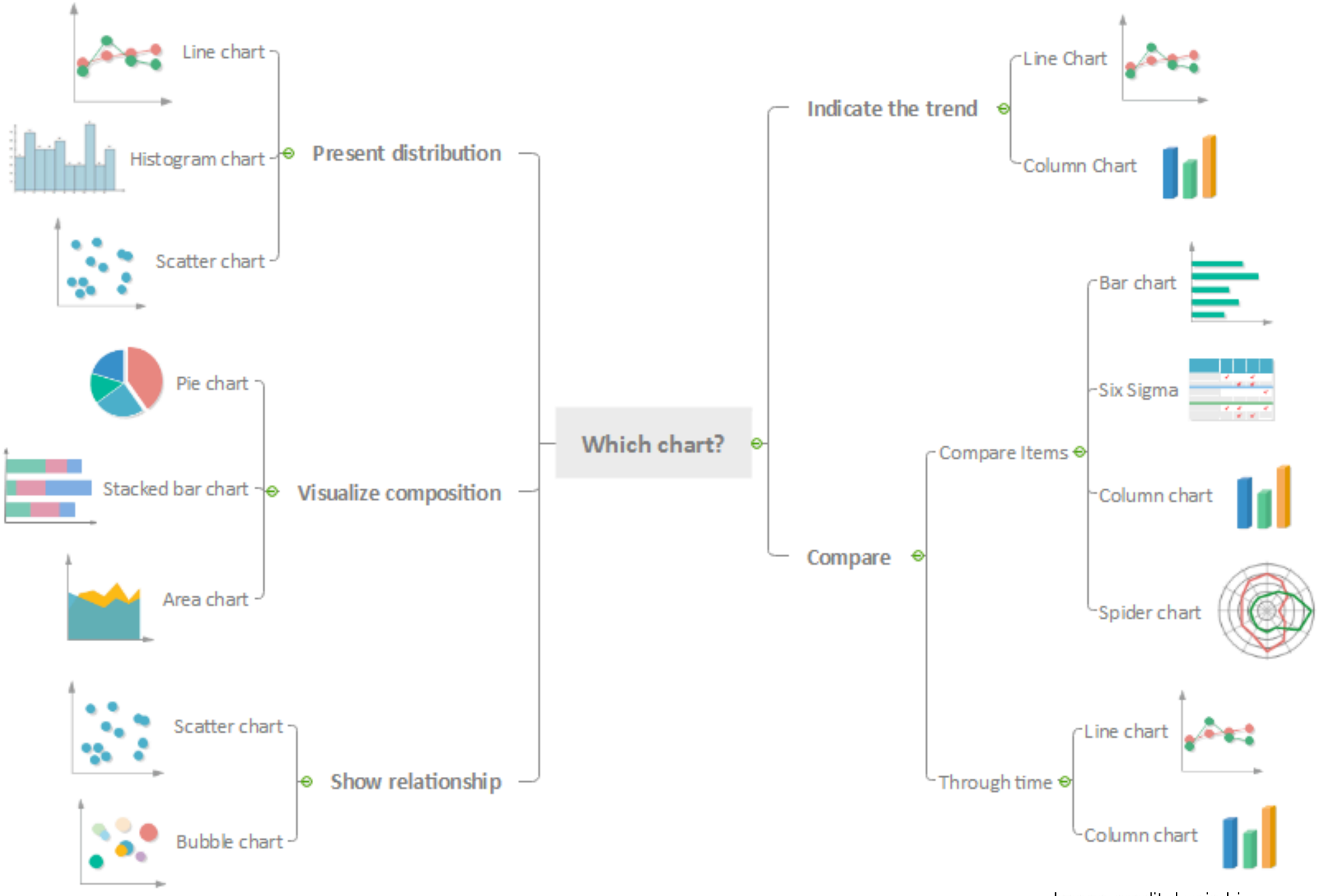


Image credit: Lynia Li

Good website resources that we can use to help us choose the right graphic forms

The Data Visualisation Catalogue

About • Blog • Shop • Resources

中文 Español Русский Türkçe

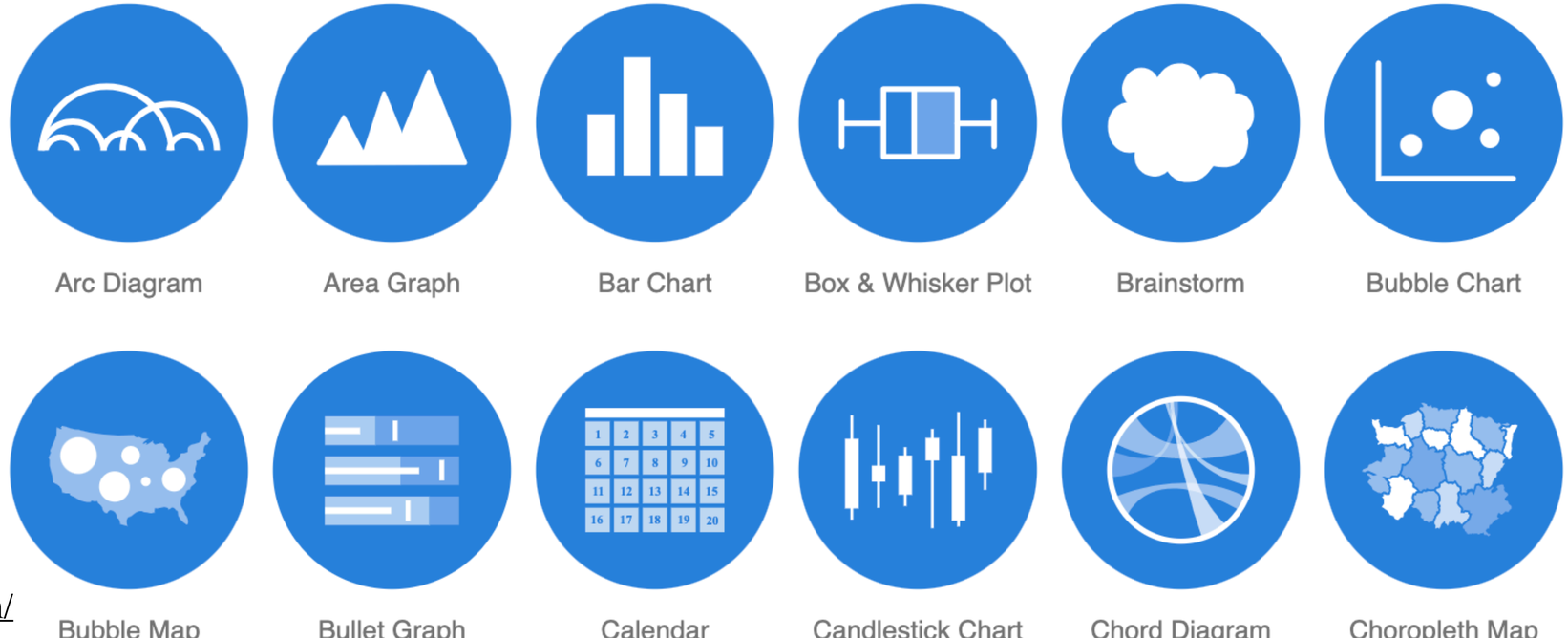
Data Visualization

OPEN

✕ 广告

Search by Function

View by List

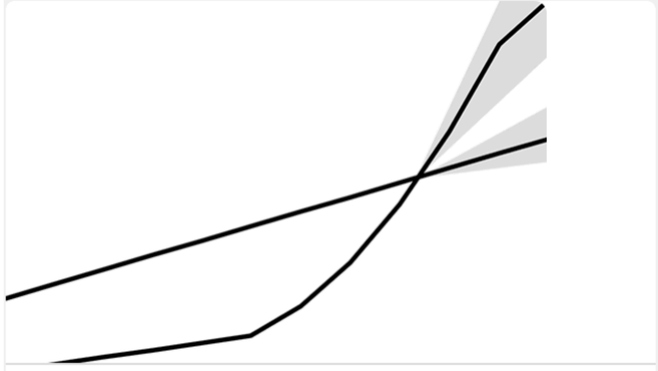


https://datavizcatalogue.com/

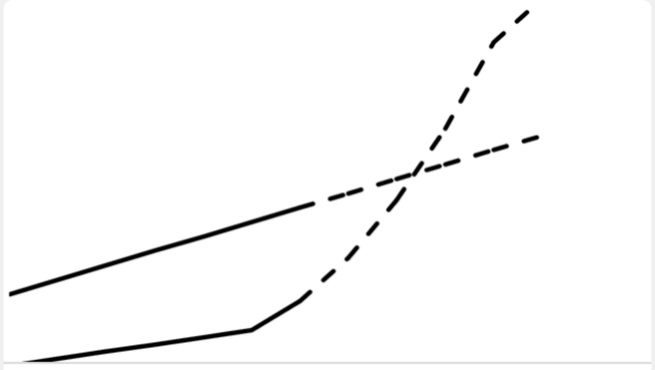
Interactive Chart Chooser

Wondering which type of graph is the best fit for your data? Explore our interactive chart chooser using the filters. For example, if you click on 3+ Points in Time, you'll see familiar faces like line graphs and meet new friends like multimedia timelines.

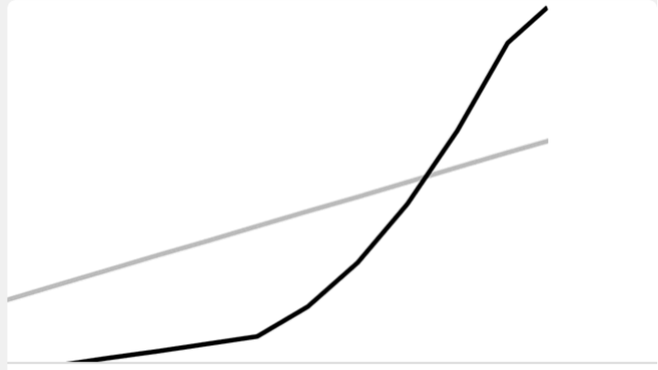
All | 1 Point in Time | 2 Points in Time | 3+ Points in Time | Comparisons | Correlation | Distribution | Exploratory | Part to Whole | Progress Towards Goals | Relationships



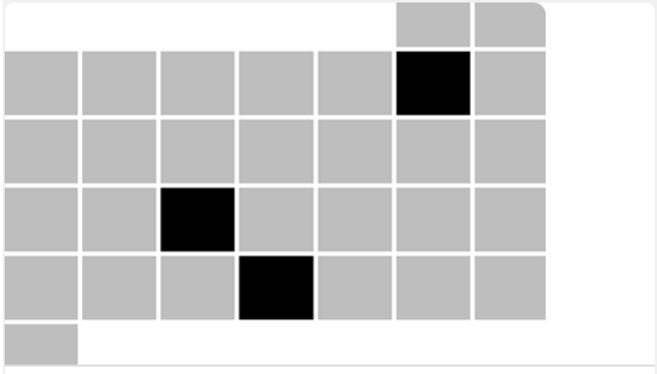
Fan Chart



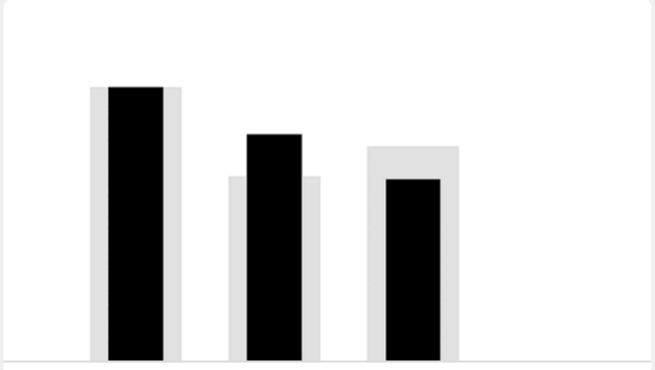
Estimations



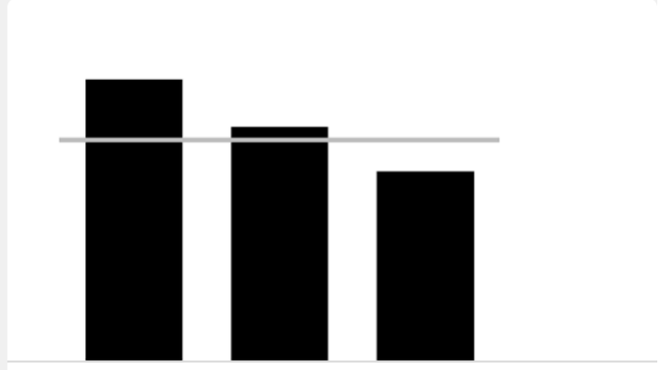
Target Lines on Line Charts



Calendars



Overlapping Bars



Target Lines on Bar Charts

https://depictdatastudio.com/charts/

Visual Vocabulary

Designing with data

There are so many ways to visualise data – how do we know which one to pick? Click on the coloured categories below to decide which data relationship is most important in your story, then look at the different types of chart within the category to form some initial ideas about what might work best. This list is not meant to be exhaustive, nor a wizard, but is a useful starting point for making informative and meaningful data visualisations

Inspired by the Graphic Continuum by Jon Schwabish and Severino Ribecca

Deviation | Correlation | Change v Time | Ranking | Distribution | Part to whole | Magnitude | Spatial | Flow

Distribution

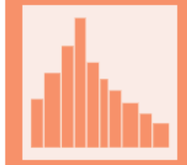
Show values in a dataset and how often they occur. The shape (or 'skew') of a distribution can be a memorable way of highlighting the lack of uniformity or equality in the data

Examples of use

Income distribution, population (age/sex) distribution

Chart types

histogram



The standard way to show a statistical distribution - keep the gaps between columns small to highlight the 'shape' of the data.

boxplot

Summarise multiple distributions by showing the median (centre) and range of the data

violin

Similar to a box plot but more effective with complex distributions (data that cannot be summarised with simple average).

population-pyramis

A standard way for showing the age and sex breakdown of a population distribution; effectively, back to back histograms

dot-plot-strip

Good for showing individual values in a distribution, can be a problem when too many dots have the same value

dot-plot

A simple way of showing the range (min/max) of data across multiple categories.

Financial Times
《金融时报》

https://ft-interactive.github.io/visual-vocabulary/

可视化辞典

Visual vocabulary

数据的設計 *Designing with data*

有许多方法可以把数据可视化，但我们怎么知道要挑选哪一种？使用本表顶部的类别，来决定在你的故事中，哪一种数据关系最重要。接着选择类别下方不同种类的图表，找出最适合表达的图表。这份列表并不是详尽无遗的，也不是一份万能向导，而是一个有用的制作起点，能協助你设计出内容丰富、有意义的数据可视化。

FT graphic: Alan Smith, Chris Campbell, Ian Bott, Liz Faunce, Graham Parrish, Billy Ehrenberg, Paul McCallum, Michael Stabe
Inspired by The Graphic Continuum by Jon Schwabish and Severino Ribecca
中文版设计制作：史书华·万丹 · Jane Wang

ft.com/vocabulary

离差
Deviation

强调相对于一个固定参考值的变化（正/负值）。通常参考值为零，但也可能是一个百分数据或是长期平均值。它能用来展现总度量与（正向/中立/负面）。

FT使用范例：贸易顺差/逆差、气候变化

分向条形图 Divergingbar

一种简单、标准化的条形图，能同时处理正/负数值。

分向堆叠条形图 Diverging stacked bar

最适合用来展现单纯的态度（正向/中立/负面）的调查数据。

成对条形图 Spine

将单一数值比较两组对比的相关（例如男性/女性）。

盈余/赤字填充线图 Surplus/deficit filled line

同时部分曲线表达一种平衡关系，或者起伏代于某个基准线，或者起伏相差距之间。

相关性
Correlation

展示两个或多个变量之间的关系，要注意的是，跟所有预测说明，许多读者会认为你用展示的两个变量之间存在因果关系（例如一个变量导致另一个变量变化）。

FT使用范例：通胀与失业率、收入与平均寿命

散点图 Scatterplot

最常用来展示连续变量关系的标准方式，每一个变量有自己的轴线。

折线图+柱状图 Line + Column

常用来一个数量（柱状图）和一个比率（折线图）的关系。

链接散点图 Connected scatterplot

通常用来展示两个变量的关系如何随着时间而变化。

泡泡图 Bubble

为最有趣而制作，但你要清楚三维变量和二维变量的大小，从而准确更多细节。

XY热图 XY heatmap

适合展示两组不同类别数据之间的关系，但不适合展示数据的确切数值。

排序
Ranking

当某个项目在排序列表中的位置比其他对数值或相对数值的大小更重要时，使用这种图表。不要将他强调测量关注的重点。

FT使用范例：财富、损失、排名表、选区选举结果

排序条形图 Ordered bar

按照顺序排列对、标准条形图更容易显示数值的趋势。

排序柱状图 Ordered column

同上。

排序比例符号 Ordered proportional symbol

为展示有巨大差异的数值时，或者无需表现数量之间的精确差异时，使用这类图表。

点状条纹图 Dot strip plot

适合展示连续上趋势，或者变换密度时，展现点多重聚的趋势。

斜度图 Slope

最适合用来展示趋势随时间一段时间之间的变化，或在不同类别间的变化。

棒棒糖图 Lollipop

棒棒糖图比标准的条形图相对更简洁明引人注目。注意数值差，也能表现表示数量的顺序和大小。

凹凸图 Bump

可以很好地展示随着在多个时期间的变化，对于大的数据集，可考虑用颜色将结果分组。

分布
Distribution

显示数据集中的数值及其出现的频率，分布的形状（或偏离程度）是突出数据的不一致或不平均的方便记忆的方式。

FT使用范例：收入分配、人口（年龄/性别）分布

直方图 Histogram

绘计分布的标准呈现方式，要确小每一个柱状之间的间隔，以凸显数据呈现出的"形状"。

箱线图 Boxplot

通过中位数（中值）、数据范围的呈现，来展现多个数量分布。

小提琴图 Violin plot

类似于箱线图，但可以更直观地呈现数据的分布（当数量大法则效果的平均数未偏离）。

人口金字塔 Population pyramid

展示人口结构在时间分布的标准方式，实际上是横对称的直方图。

点状条纹图 Dot strip plot

用来呈现分布中个别数值的好方法，但某次多因点数有相同大小的情况下呈现。

点状图 Dot plot

呈现不同类别数据的变化或范围（最小值/最大值）。

二维条纹图 Barcode plot

常用在对象图形，适合在一张图中呈现所有数据，这种图表最适合用来凸显个别数值。

累积曲线图 Cumulative curve

呈现数据分布不均等的好方法，Y轴的高度是累积数据，X轴的高度是态度。

频率多边图 Frequency polygons

使用时呈现多个数据分布，类似于直方图的折线图，最好一次只展现3—4组数据。

随时间的变化
Change over Time

强调随时的变化，有可能是短期（一日内）波动或长到数十年或数百年的改变。为了向读者提供恰当的背景信息，选择正确的时间段很重要。

FT使用范例：股价变动、经济动态时间序列

折线图 Line

用来显示时间序列变化趋势的标准方式，如果数据不规则，可以考虑用记号来表示数据点。

柱状图 Column

适合用来展示随时间变化的趋势，但要保留时一次只呈现一个数列。

折线图+柱状图 Line + column

适合用来呈现数量（柱状）和比率（折线）随时间的关系变化。

股价 Stock price

通常用来于数据的每日动向，由呈现每日的开盘、收盘价、以及高点和低点。

斜度 Slope

只要数据能降低为2到3个有用不同类别信息，这图表形式会用来展现数据的变化。

区域图 Area chart

要小心使用，适合展示整体的变化，但要注意当线条交叉时。

扇形图（预测） Fan chart (projection)

用来呈现未来预测值的不确定性，通常呈现随着不确定性增大。

链接散点图 Connected scatterplot

适合用来展示两个变量的数据变化，只要数据有明显的趋势。

日历式热图 Calendar heatmap

适合用来呈现时间顺序（每日、每周、每月）中的数据趋势模式的展示。

普利斯特利时间轴 Priestley timeline

适用于日期和时间长度超过数年的长期数据，以此纪念英国化学家约瑟夫·普里斯特利。

圆圈时间轴 Circle timeline

适合用来不同类别数据的不连续数据（例如不同人物的出生日期）。

垂直时间轴 Vertical timeline

以Y轴表示时间，在每项设置上面的图表不同时间，呈现出时间的时间序列。

震波图 Seismogram

因图表时间的替代选择，适合展示数量大幅变动的图表。

规模
Magnitude

用来比较数据的规模，有可能是比较相对规模（显示出哪一个比较大），有可能是比较绝对规模（需要显示出精确的差异）。通常用来比较数量（例如钱、人、美元），而不是相比计算后的比率或百分比。

FT使用范例：大宗商品产量、市值

柱状图 Column

用来比较数据的标准图表，最适合用在数值小于为0。

条形图 Bar

同上，适用于不超过两序列的数据，以及能容纳标签的数据。

成对柱状图 Paired column

类似于一般柱状图，但是用于比较两个量的同一为数的相对比较（例如男和女）。

成对条形图 Paired bar

同上。

比例堆叠条形图 Marimekko

适合用来同时呈现数量的大小与占比，只要数量不超太复杂。

比例符号图 Proportional symbol

为展示巨大差异的数值时，或者无需表现数量之间的精确差异时，使用这类图表。

象形符号 Isotype (pictogram)

特别适用于表现趋势，但只适用于数量（不要使一条手臂表示小数）。

棒棒糖图 Lollipop

比较清楚地表现相对差异，但只适用引人注目的数量，但如果不需要为0，也最好为0。

雷达图 Radar

呈现多变量的一个节省空间的方式，但要必要变量的顺序对表现有有一定的逻辑。

平行坐标图 Parallel coordinates

要这图的替代选择，用于，变量的顺序很重要，同时呈现数量和单图表情况。

子弹图 Bullet

适合以目标值或区间值为参数展示变量。

分组符号图 Grouped symbol

柱状图，集合图的替代选择，相同类别数据呈现出不同数量，要比较不同类别。

部分和整体的关系
Part-to-whole

能显示出一个整体如何被拆解成不同组成。如果读者只是想了解个别组成部分的大小，不能选用规模类的图表。

FT使用范例：财政预算、公司架构、全国选举结果

堆叠柱状图 Stacked column

可以表现整体图表形分为整体的关系，但如果组成部分过多会造成难解阅读。

比例堆叠条形图 Marimekko

适合用来同时呈现数量的大小与占比，只要数量不超太复杂。

饼图 Pie

用来呈现部分与整体关系的常见方式，但要注意的是，这种图表容易被人解读出部分的大小。

甜甜圈图 Donut

类似于饼图，但中间的空间能加入更多信息（例如总数大小）

树状图 Treemap

用来呈现数据层关系的部分与整体的图表，但如果组成太细碎，会使用图表难以理解。

沃罗诺伊图 Voronoi

把地图或区域的图表，每个区域的任何一点，离距离本区域的中心点的距离比其他区域的中心点更近。（沃罗诺伊图以俄罗斯数学家）

扶形图 Arc

一个半圆，通常用来呈现设计的数据结果。

网格Gridplot

适合用来呈现所谓比较表，最适合以多翻格的形式呈现图表。

文氏图 Venn

通常只用于简化的数据呈现。

瀑布图 Waterfall

为某些相关部分为负值时，适合用来表现呈现部分和整体的关系。

地理空间
Spatial

当数据中的精确位置和地理分布随着比其他信息对读者来说更重要时，可使用这类图表。

FT使用范例：人口密度、自然资源分布、自然灾害风险/影响、集水区域、投票结果差异

基础地理分布图（比率/比值）
Basic choropleth (ratio/value)

把数据按照地图上展示的标准方法，但在呈现的最比值不是绝对数值，同时要使用一个合理的基础范围。

比例符号地图（数量/规模）
Proportional symbol (count/magnitude)

呈现绝对数值不是比值，要留意的是，数量的规模差异容易呈现。

流向地图 Flowmap

在地图上展示有明确移动方向的信息。

等高线地图 Contour map

在地图上展示有相同数值的区域，除使用海拔色彩来显示此区，负数值。

均等化示意地图 Equalised cartogram

把地图上的单位区域转化为相同大小的相同图形，适合以相等的数值代表区域色区。

缩放统计地图（数值）Scaled cartogram (value)

放大或缩小地图，使得每一个区域根据数值决定大小。

点状密度地图 Dot density

用来展示单一事物的地点，还用来注意者位置注意的频率。

热度 地图 Heatmap

通过强烈的颜色来表示区域数量，类似于基础地理分布图，但是地图不会被明确地划分区域来表现。

流向
Flow

向读者展示两个或两个以上的状态、情况之间的流动量或流动强度、这里的状态、情况可能是使用关系或地理位置。

FT使用范例：资金、贸易、移民、资讯、谈论的流向、关系图表。

桑基图 Sankey

呈现从一个情况到至少另一个情况的流向变化，适合能展现流动相对量的效果。

瀑布图 Waterfall

通过流动过程来展示数量的顺序变化，最典型的是展示预算流动，可以用负区域为负数或数分。

和弦图 Chord

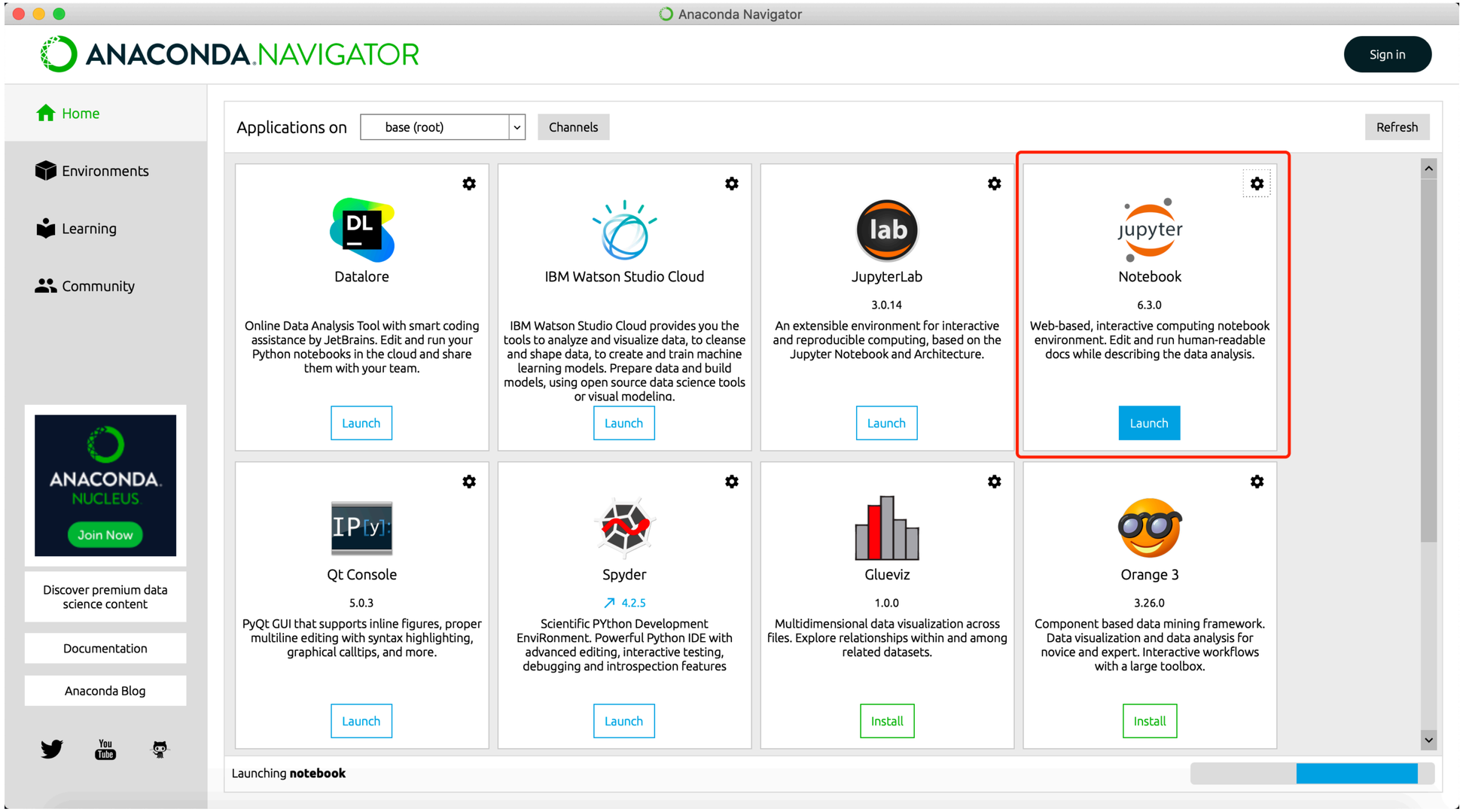
一种复杂的有表现力的图表，适合在两种不同对象之间有交流（以及每个表现最强）。

网络图 Network

在不同类型的对象之间，展示出关系强度和方向关联关系。

LAB TIME

Recap


Anaconda Navigator
ANACONDA.NAVIGATOR
Sign in
Home
Environments
Learning
Community
Applications on
base (root)
Channels
Refresh
Datalore
Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.
Launch
IBM Watson Studio Cloud
IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.
Launch
JupyterLab
3.0.14
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.
Launch
Notebook
6.3.0
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.
Launch
ANACONDA. NUCLEUS
Join Now
Discover premium data science content
Documentation
Anaconda Blog
Qt Console
5.0.3
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
Launch
Spyder
4.2.5
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features
Launch
Glueviz
1.0.0
Multidimensional data visualization across files. Explore relationships within and among related datasets.
Install
Orange 3
3.26.0
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.
Install
Launching notebook

Python basics

Basic data types in Python

- Integer
- Float
- String
- Boolean

Basic data structures in Python

- List and Tuple
- Dictionary and Set

Tuple and List

- **Tuples are arrays enclosed with round brackets( )** for storing multiple variables
- Variables with data types can be stored into a Tuple
- A Tuple operates like a string and therefore can be indexed from the beginning (positive) and the end (negative)
- Tuples are immutable
- In order to manipulate a tuple, a new one has to be created
- Tuples can be nested e.g. Tuple2 = (1,2,(3,4),5)
- Due to its rigidity, **List is more commonly used than Tuple**.
- **Lists are like Tuples but are mutable and enclosed with square brackets[ ]**
- List, similar to a Tuple, operates like a string when it comes to access individual element within the data structure

Tuple and List

```
In [22]: tuple1 = (1,1.32,2,'CUC')
         print(tuple1)
         list1 = [1,1.32,2,'CUC']
         print(list1)

         (1, 1.32, 2, 'CUC')
         [1, 1.32, 2, 'CUC']
```

```
In [23]: myList = [1,2,3,4,5,1.5]
         print(sorted(myList))

         [1, 1.5, 2, 3, 4, 5]
```

```
score_list = [100,95,85,60,60.5,70]
print(score_list)
print(score_list[3])
score_list.append(50)
print(score_list)
del(score_list[6])
print(score_list[3:7])
print(score_list)
score_list.append([1,2,3,"I am from CUHK.",True])
print(score_list)
second_list = [11,22,33]
second_list.append([44,55,66])
score_list.append(second_list)
print(score_list)
score_list.remove(100)
print(score_list)
```

```
[100, 95, 85, 60, 60.5, 70]
60
[100, 95, 85, 60, 60.5, 70, 50]
[60, 60.5, 70]
[100, 95, 85, 60, 60.5, 70]
[100, 95, 85, 60, 60.5, 70, [1, 2, 3, 'I am from CUHK.', True]]
[100, 95, 85, 60, 60.5, 70, [1, 2, 3, 'I am from CUHK.', True], [11, 22, 33, [44, 55, 66]]]
[95, 85, 60, 60.5, 70, [1, 2, 3, 'I am from CUHK.', True], [11, 22, 33, [44, 55, 66]]]
```

Get more info about Python lists from google developer website:
https://developers.google.com/edu/python/lists

Set and Dictionary

- Similar to lists and tuples, sets support different Python types
- **Sets use {} (braces) to embed values/elements**
- Sets do not allow duplicates
- Lists can be converted into sets with the set function
- **Dictionary store data in an array of key-value pairs in braces**
- For instance, here is a dictionary instance: dict = {“key1”:1,”key2”:2,”key3”:3}
- 1st column representing the key and 2nd column representing the value
- <dict_name>.keys() returns all the keys
- <dict_name>.values.() returns all the values

Set and Dictionary

```
In [30]: dict1 = {'Peter':80, 'David':90, 'Mary':100}
         print(dict1)
         print(dict1.keys())
         print(dict1.values())
         print(dict1["David"])
         score_list = []
         score_list.append(dict1)
         print(score_list)

         {'Peter': 80, 'David': 90, 'Mary': 100}
         dict_keys(['Peter', 'David', 'Mary'])
         dict_values([80, 90, 100])
         90
         [{'Peter': 80, 'David': 90, 'Mary': 100}]
```

```
In [31]: test1 = {"Peter":50,"David":60,"Mary":70}
         assignment1 = {"Peter":50,"David":70,"Mary":80}
         score_list = []
         score_list.append(test1)
         score_list.append(assignment1)
         print(score_list)

         [{'Peter': 50, 'David': 60, 'Mary': 70}, {'Peter': 50, 'David': 70, 'Mary': 80}]
```

```
In [32]: # set of mixed datatypes
         my_set = {1.0, "Hello", (1, 2, 3)}
         print(my_set)
         print(type(my_set))
         # set cannot have duplicates
         my_set = {1, 2, 3, 4, 3, 2}
         print(my_set)

         {1.0, 'Hello', (1, 2, 3)}
         <class 'set'>
         {1, 2, 3, 4}
```

Basic operations in Python

- Variable Assignment
- Mathematical Operations
- Functional Decomposition and Abstraction
- Logical Operations
- Looping Operations

Looping operations

```
In [33]: i = 1
         for i in range(1,10):
             print(i)

         1
         2
         3
         4
         5
         6
         7
         8
         9
```

```
In [36]: hrs_list = [30.0,40.0,50.0,60.0]
         rate_list = [65.0,75.0,65.0,75.0]
         name_list = ['John',"Mike","Mary","Jane"]
         fee_list = []
         for number in range(len(name_list)):
             hrs = hrs_list[number]
             rate = rate_list[number]
             fee = hrs * rate
             fee_list.append(fee)
             money_made = name_list[number] + " makes " + str(fee) + "."
             print(money_made)
         money_list = []
         money_list.append(name_list)
         money_list.append(fee_list)
         print(money_list)

         John makes 1950.0.
         Mike makes 3000.0.
         Mary makes 3250.0.
         Jane makes 4500.0.
         [['John', 'Mike', 'Mary', 'Jane'], [1950.0, 3000.0, 3250.0, 4500.0]]
```

Conditional operations

Pay attention to the indent (缩进).

```
In [38]: my_list = ['red','green','blue','orange','black']
         index = 0
         for i in my_list:
             if i == "blue":
                 print(index, "Blue is printed.")
                 index +=1
             else:
                 print(index, my_list[index])
                 index = index + 1
         print("This is done.")
```

```
0 red
1 green
2 Blue is printed.
3 orange
4 black
This is done.
```

i++, which we are familiar with in Javascript, could not work in **Python** since it does not have unary increment/decrement operator(++/--). Instead to increment a value, use a **+= 1**. to decrement a value, use– a -= **1**.

Introduction to Pandas

Pandas

Pandas, Python Data Analysis Library, is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

- Pandas allows us to deal with 2 data structures: series and data frame
- In our class, we'll only concentrate on data frame as it is more commonly used
- A data frame is consisted of rows and columns. It can be made from Python list and dictionary objects
- Pandas can put a dictionary of list into a data frame
- There are build-in Pandas functions for reading (e.g. df=pd.read_csv('<file name>')) and writing to CSV files (e.g df.to_csv('<file name>', sep='\t', encoding='utf-8'))

See the official documentation here: https://pandas.pydata.org/docs/

Pandas

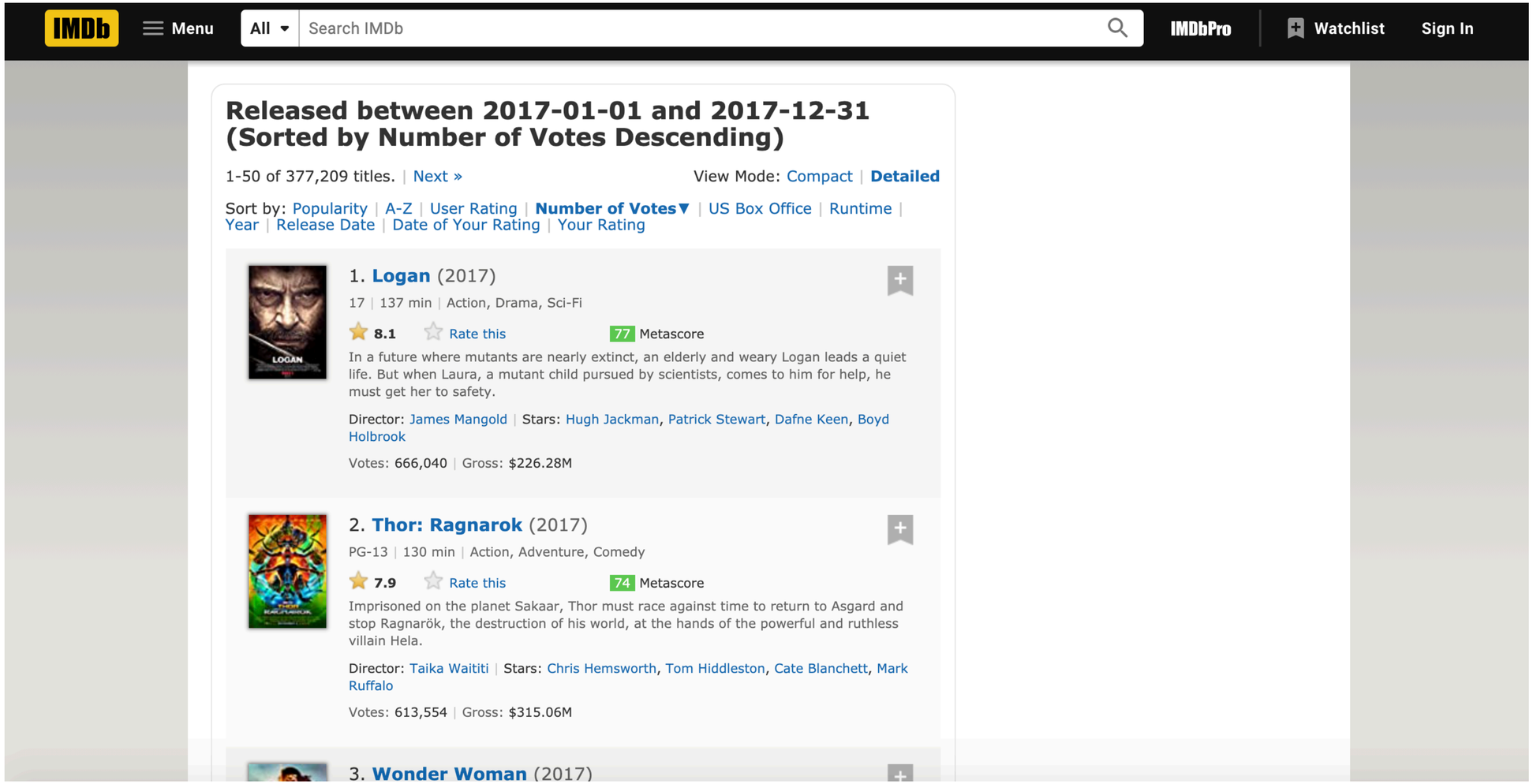
```
import pandas as pd
df1 = pd.DataFrame({
    # Define dataframe as a dictionary object
   'Product ID': [1, 2, 3, 4],
    # add Product Name and Color here
   'Product Name': ['t-shirt','jeans','shirt','skirt'],
   'Color':['blue','green','red','black'],
   'Units Sold':[250,300,180,200]
})
print(df1)
```

```
In [49]: import pandas as pd
         df1 = pd.DataFrame({
              # Define dataframe as a dictionary object
             'Product ID': [1, 2, 3, 4],
              # add Product Name and Color here
             'Product Name': ['t-shirt','jeans','shirt','skirt'],
             'Color':['blue','green','red','black'],
             'Units Sold':[250,300,180,200]
         })
         print(df1)

   Product ID Product Name  Color  Units Sold
0           1      t-shirt   blue         250
1           2        jeans  green         300
2           3        shirt    red         180
3           4        skirt  black         200
```

With a little bit background in Python, it’s now time to scrape!

HOW TO
SCRAPE WEBPAGES
WITH
BEAUTIFUL
SOUP

Using IMDB as example



Source: https://www.imdb.com/search/title/?release_date=2017&sort=num_votes,desc&page=1

Scraping Data with Beautiful Soup and Storing the Result as CSV File

Collecting and Parsing a Web Page

In [*]:
```
import requests
from bs4 import BeautifulSoup
```

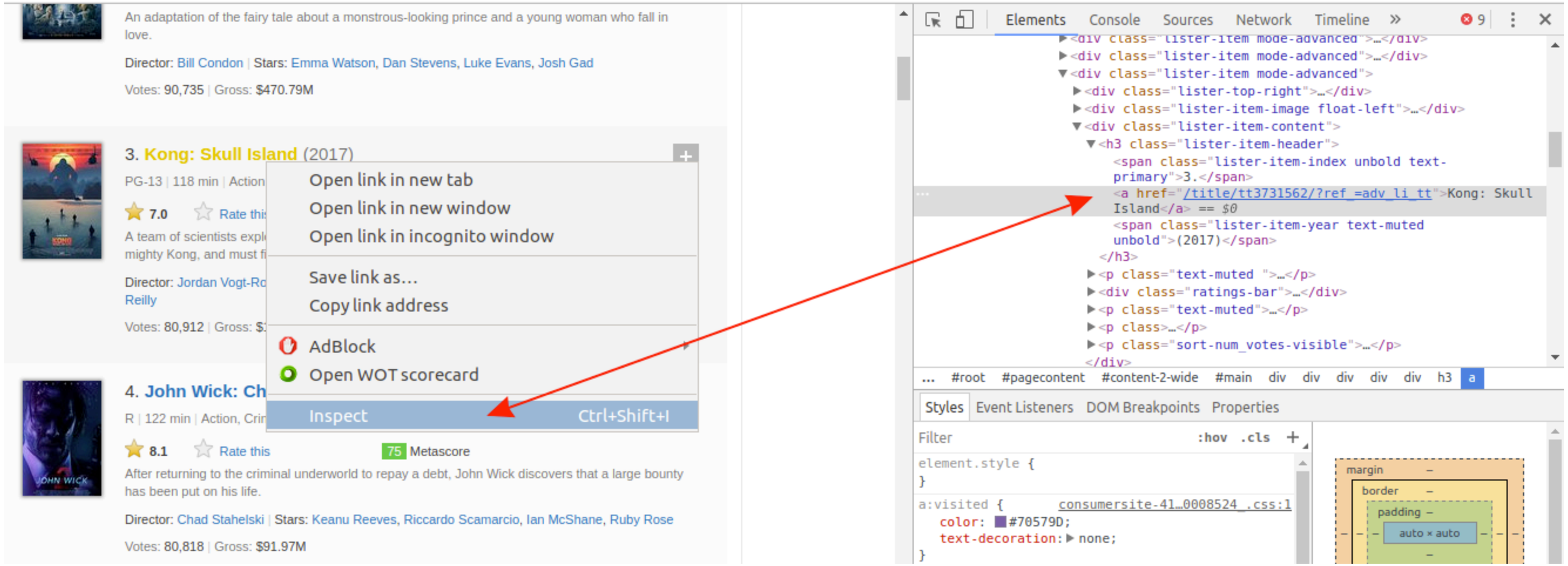
*** means this line of code is still running at the moment**

```
# Collect first page of artists' list
page = requests.get('https://web.archive.org/web/20121007172955/https://www.nga.gov/collection/anZ1.htm')

# Create a BeautifulSoup object
soup = BeautifulSoup(page.text, 'html.parser')
```

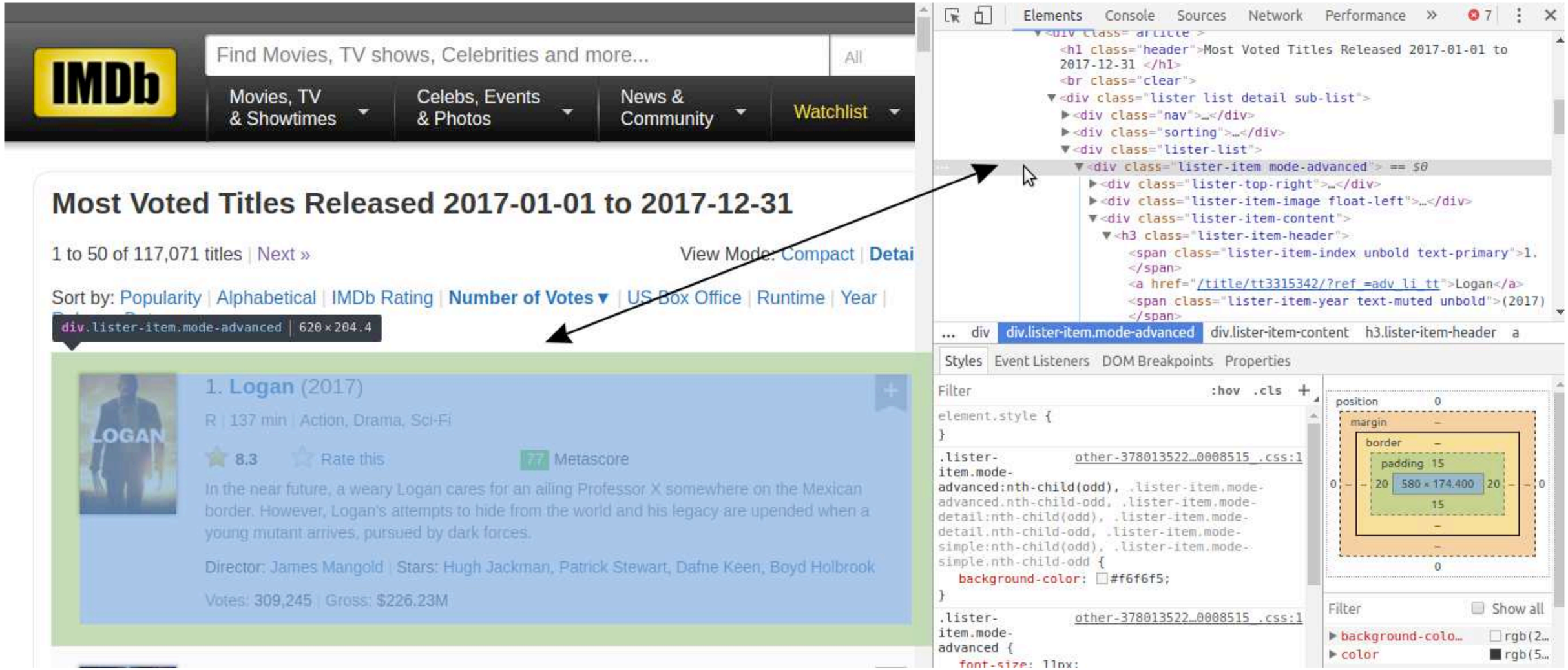
In [ ]:

(This is **NOT** the IMDB example)

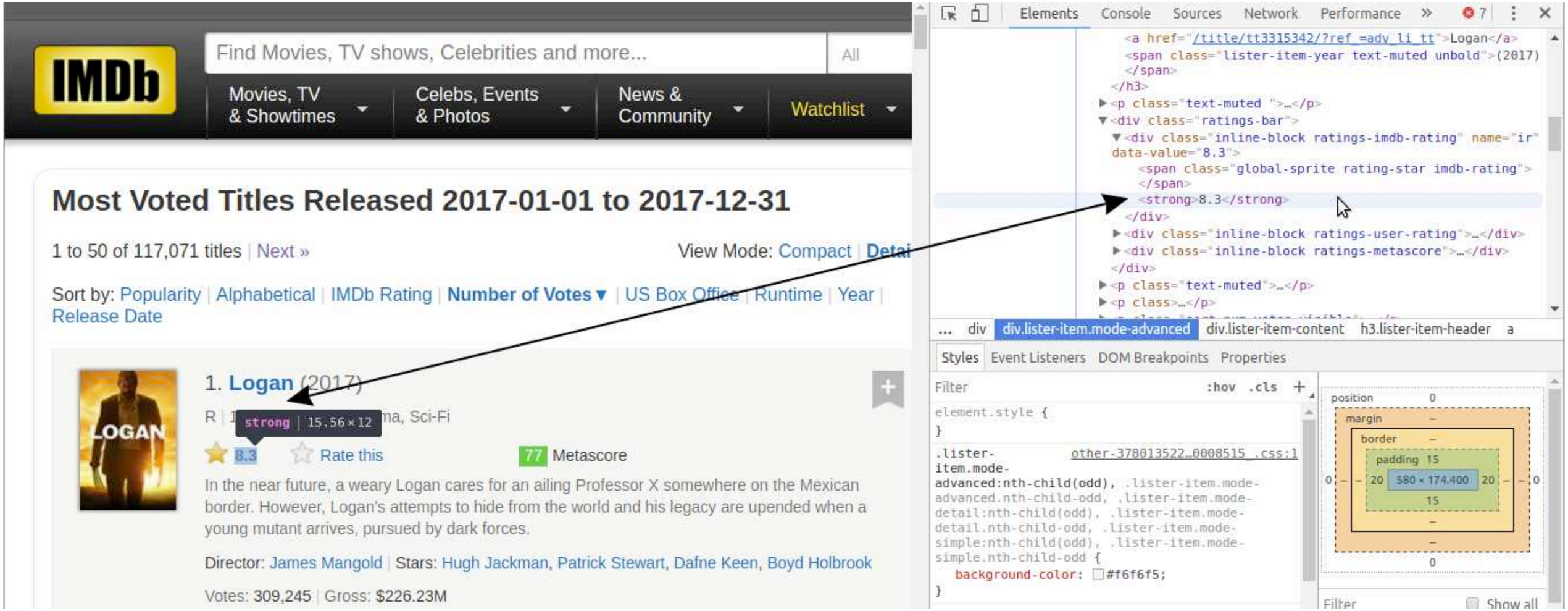
Understanding the HTML structure of a single page first



All of the information for each movie, including the poster, is contained in a `div` tag



Within each movie nested tags we'll find the information we need, like a movie's rating.

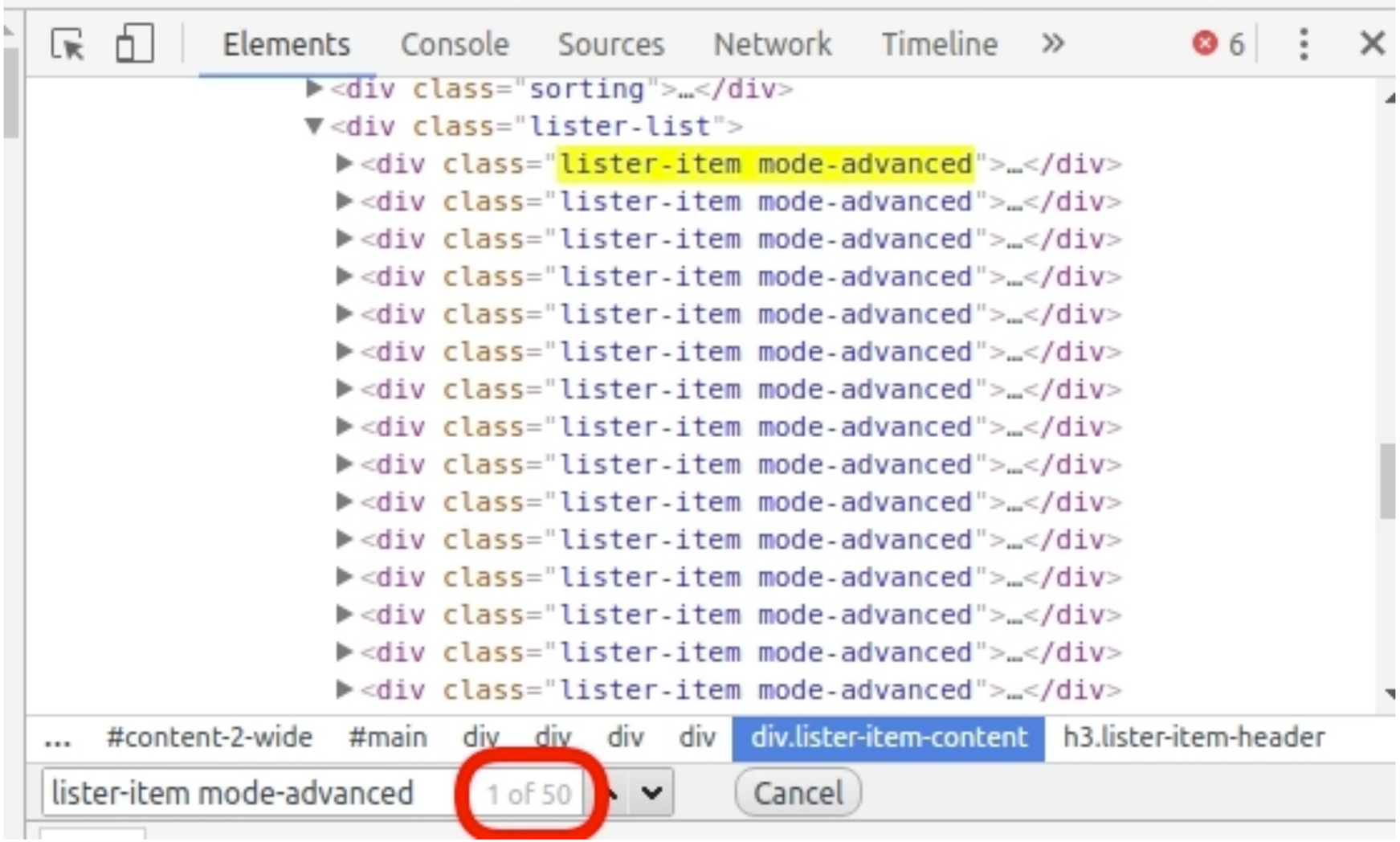


There are 50 movies shown per page, so there should be a `div` container for each. Let's extract all these 50 containers by parsing the HTML document from our earlier request.

Step 1: requesting the content of the web page and using BeautifulSoup to parse the HTML content

In [1]:
```
import requests
from bs4 import BeautifulSoup


page = requests.get('https://www.imdb.com/search/title?release_date=2017&sort=num_votes,desc&page=1')

# Create a BeautifulSoup object
soup = BeautifulSoup(page.text, 'html.parser')
print(soup)
```

```
<!DOCTYPE html>

<html xmlns:fb="http://www.facebook.com/2008/fbml" xmlns:og="http://ogp.me/ns#">
<head>
<meta charset="utf-8"/>
<meta content="IE=edge" http-equiv="X-UA-Compatible"/>
<meta content="app-id=342792525, app-argument=imdb:///?src=mdot" name="apple-itunes-app"/>
<script type="text/javascript">var IMDbTimer={starttime: new Date().getTime(),pt:'java'};</script>
<script>
    if (typeof uet == 'function') {
      uet("bb", "LoadTitle", {wb: 1});
    }
</script>
<script>(function(t){ (t.events = t.events || {})["csm_head_pre_title"] = new Date().getTime(); })(IMDbTimer);</scrip
t>
<title>Released between 2017-01-01 and 2017-12-31
(Sorted by Number of Votes Descending) - IMDb</title>
<script>(function(t){ (t.events = t.events || {})["csm_head_post_title"] = new Date().getTime(); })(IMDbTimer);</scri
```



quick search (Ctrl + F)
There are 50 containers which all have distinctive class "lister-item mode-advanced"

Step 2: Pulling Text From a Web Page

Now let's use the find_all() method to extract all the div containers that have a class attribute of ***lister-item mode-advanced***:

```
In [2]: movie_containers = soup.find_all('div', class_ = 'lister-item mode-advanced')
        print(type(movie_containers))
        print(len(movie_containers))

        <class 'bs4.element.ResultSet'>
        50
```

Now we'll select only the first container, and extract, by turn, each item of interest:

- The name of the movie.
- The year of release.
- The IMDB rating.
- The number of votes.

Extracting the data for the first movie

```
In [20]: first_movie = movie_containers[0]
         first_movie
```

```
Out[20]: <div class="lister-item mode-advanced">
<div class="lister-top-right">
<div class="ribbonize" data-caller="filmosearch" data-tconst="tt3315342"></div>
</div>
<div class="lister-item-image float-left">
<a href="/title/tt3315342/"> <img alt="Logan" class="loadlate" data-tconst="tt3315342" height="98" loadlate="https://
m.media-amazon.com/images/M/MV5BYzc5MTU4N2EtYTkyMi00NjdhLTg3NWEtMTY4OTEyMzJhZTAzXkEyXkFqcGdeQXVyNjc1NTYyMjg@._V1_UX67
_CR0,0,67,98_AL_.jpg" src="https://m.media-amazon.com/images/S/sash/4FyxwxECzL-U1J8.png" width="67"/>
</a> </div>
<div class="lister-item-content">
<h3 class="lister-item-header">
<span class="lister-item-index unbold text-primary">1.</span>
<a href="/title/tt3315342/">Logan</a>
<span class="lister-item-year text-muted unbold">(2017)</span>
</h3>
<p class="text-muted">
<span class="certificate">17</span>
<span class="ghost">|</span>
<span class="runtime">137 min</span>
<span class="ghost">|</span>
<span class="genre">
Action, Drama, Sci-Fi            </span>
</p>
<div class="ratings-bar">
<div class="inline-block ratings-imdb-rating" data-value="8.1" name="ir">
<span class="global-sprite rating-star imdb-rating"></span>
<strong>8.1</strong>
</div>
<div class="inline-block ratings-user-rating">
<span class="userRatingValue" data-tconst="tt3315342" id="urv_tt3315342">
```

As you can see, the HTML content of one container is quite long!

Let’s go back to the DevTools , learn the structure and extract the information we want.

```
▼<div class="lister-list">
  ▼<div class="lister-item mode-advanced">          ← 1st container
    ▶<div class="lister-top-right">…</div>          ← 1st div
    ▶<div class="lister-item-image float-left">…    ← 2nd div
    </div>
    ▼<div class="lister-item-content">              ← 3rd div
      ▼<h3 class="lister-item-header">              ← <h3>
        <span class="lister-item-index unbold text-
        primary">1.</span>
        <a href="/title/tt3315342/?ref_=adv_li_tt">  ← <a>
        Logan</a> == $0
        <span class="lister-item-year text-muted
        unbold">(2017)</span>
      </h3>
      ▶<p class="text-muted ">…</p>
      ▶<div class="ratings-bar">…</div>
      ▶<p class="text-muted">…</p>
      ▶<p class>…</p>
      ▶<p class="sort-num_votes-visible">…</p>
    </div>
  </div>
  ▶<div class="lister-item mode-advanced">…</div>
  ▶<div class="lister-item mode-advanced">…</div>
  ▶<div class="lister-item mode-advanced">…</div>
  ▶<div class="lister-item mode-advanced">…</div>
```

The other movie containers

Extract the name of the movie:

```
In [21]: first_name = first_movie.h3.a.text
         first_name

Out[21]: 'Logan'
```

```
▼<h3 class="lister-item-header">
  <span class="lister-item-index unbold text-primary">1.</span>
  <a href="/title/tt3315342/?ref_=adv_li_tt">Logan</a>
  <span class="lister-item-year text-muted unbold">(2017)</span>
</h3>
```

1st → 1.

2nd → (2017)

distinctive mark

Dot notation will only access the first `span` element. We'll search by the distinctive mark of the second `<span>`. We'll use the `find()` method which is almost the same as `find_all()`, except that it only returns the first match.

Extract the year of the movie release:

```
In [22]: first_year = first_movie.h3.find('span', class_ = 'lister-item-year text-muted unbold')
         first_year

Out[22]: <span class="lister-item-year text-muted unbold">(2017)</span>

In [23]: first_year = first_year.text
         first_year

Out[23]: '(2017)'
```

The <h3> from before

```
▼<div class="lister-item-content">
  ▶<h3 class="lister-item-header">…</h3>
  ▶<p class="text-muted ">…</p>
  ▼<div class="ratings-bar">
    ▼<div class="inline-block ratings-imdb-rating"
    name="ir" data-value="8.3">
        <span class="global-sprite rating-star
        imdb-rating"></span>
        <strong>8.3</strong> == $0
      </div>
    ▶<div class="inline-block ratings-user-
    rating">…</div>
    ▶<div class="inline-block ratings-metascore">…
    </div>
    </div>
  ▶<p class="text-muted">…</p>
```

Extract the movie rating:

```
In [24]: first_movie.strong

Out[24]: <strong>8.1</strong>

In [25]: first_imdb = float(first_movie.strong.text)
         first_imdb

Out[25]: 8.1
```

```
<span class="metascore  favorable">77
</span>
"
Metascore
"
</div>
</div>
▶<p class="text-muted">…</p>
▶<p class>…</p>
▼<p class="sort-num_votes-visible">
<span class="text-muted">Votes:</span>
<span name="nv" data-value="316536">316,536
</span> == $0
<span class="ghost">|</span>
<span class="text-muted">Gross:</span>
<span name="nv" data-value="226,252,579">
$226.25M</span>
</p>
</div>
</div>
▶<div class="lister-item mode-advanced">…</div>
▶<div class="lister-item mode-advanced">…</div>
▶<div class="lister-item mode-advanced">…</div>
```

the containing the metascore

distinctive mark

number of votes (two sources)

The `name` attribute is different from the `class` attribute. Using BeautifulSoup we can access elements by any attribute. The `find()` and `find_all()` functions have a parameter named `attrs`. To this we can pass in the attributes and values we are searching for as a dictionary:

Extract the number of votes:

```
In [26]: first_votes = first_movie.find('span', attrs = {'name':'nv'})
         first_votes

Out[26]: <span data-value="664874" name="nv">664,874</span>

In [27]: first_votes['data-value']

Out[27]: '664874'
```

That’s it! We’re now in a position to easily write a script for scraping a single page.

Step 3: The script for a single page

Now let's put together the code above:

- Declare some `list` variables to have something to store the extracted data in.
- Loop through each container in `movie_containers` (the variable which contains all the 50 movie containers).

```
In [28]: # Lists to store the scraped data in
         names = []
         years = []
         imdb_ratings = []
         votes = []
         # Extract data from individual movie container
         for container in movie_containers:
         # The name
             name = container.h3.a.text
             names.append(name)
         # The year
             year = container.h3.find('span', class_ = 'lister-item-year').text
             years.append(year)
         # The IMDB rating
             imdb = float(container.strong.text)
             imdb_ratings.append(imdb)
         # The number of votes
             vote = container.find('span', attrs = {'name':'nv'})['data-value']
             votes.append(int(vote))
```

Step 4: importing Pandas and putting the data you collected so far in a table

In [30]:
```
import pandas as pd
test_df = pd.DataFrame({'movie': names,
'year': years,
'imdb': imdb_ratings,
'votes': votes
})
print(test_df.info())
test_df
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   movie   50 non-null     object
 1   year    50 non-null     object
 2   imdb    50 non-null     float64
 3   votes   50 non-null     int64
dtypes: float64(1), int64(1), object(2)
memory usage: 1.7+ KB
None
```

Out[30]:

	movie	year	imdb	votes
0	Logan	(2017)	8.1	664874
1	Thor: Ragnarok	(2017)	7.9	612069
2	Wonder Woman	(2017)	7.4	591903
3	Guardians of the Galaxy Vol. 2	(2017)	7.6	586625
4	Star Wars: Episode VIII - The Last Jedi	(2017)	7.0	575554
5	Dunkirk	(2017)	7.8	572362
6	Spider-Man: Homecoming	(2017)	7.4	530215
7	Get Out	(I) (2017)	7.7	507567
8	Blade Runner 2049	(2017)	8.0	477760
9	It	(I) (2017)	7.3	469556
10	Baby Driver	(2017)	7.6	453782
11	Three Billboards Outside Ebbing, Missouri	(2017)	8.1	449003
12	Justice League	(2017)	6.2	413599
13	Coco	(I) (2017)	8.4	404598
14	The Shape of Water	(2017)	7.3	381523
15	John Wick: Chapter 2	(2017)	7.5	377675
16	Jumanji: Welcome to the Jungle	(2017)	6.9	329308
17	La casa de papel	(2017–)	8.3	320929

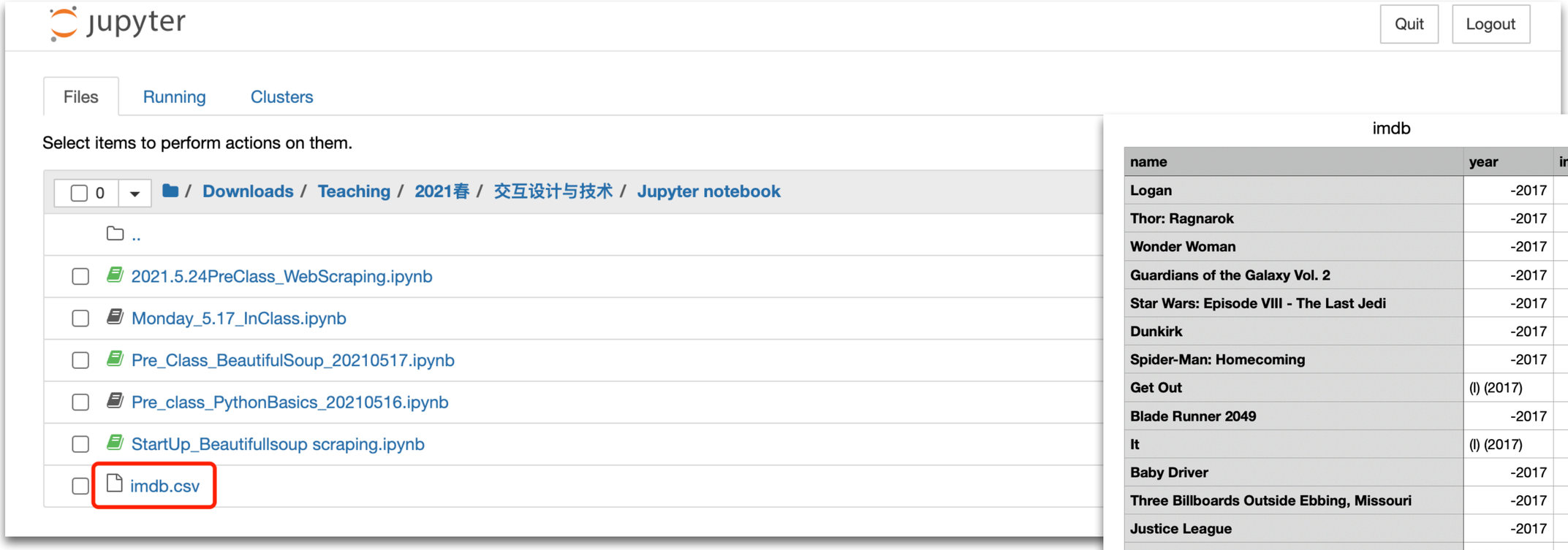
Step 5: importing Pandas and putting the data you collected so far in a table

In [10]:
```
import csv

# Lists to store the scraped data in
names = []
years = []
imdb_ratings = []
votes = []

# Create a file to write to, add headers row
f = csv.writer(open('imdb.csv', 'w'))
f.writerow(['name', 'year','imdb','vote'])

# Extract data from individual movie container
for container in movie_containers:
# The name
    name = container.h3.a.text
    names.append(name)
# The year
    year = container.h3.find('span', class_ = 'lister-item-year').text
    years.append(year)
# The IMDB rating
    imdb = float(container.strong.text)
    imdb_ratings.append(imdb)
# The number of votes
    vote = container.find('span', attrs = {'name':'nv'})['data-value']
    votes.append(int(vote))
    # Add each movie's name, year, imdb and vote data to a row
    f.writerow([name, year, imdb, vote])
```

import Python's built-in *csv* module

Next, we'll create and open a file called *imdb.csv* for us to write to (we'll use the variable *f* for file here) by using the 'w' mode. We'll also write the top row headings: name, year, imdb and vote which we'll pass to the writerow() method as a list

Run the program, no output will be returned to your terminal window. Instead, a file will be created in the directory called `imdb.csv`:



imdb

name	year	imdb	vote
Logan	-2017	8.1	666040
Thor: Ragnarok	-2017	7.9	613554
Wonder Woman	-2017	7.4	593099
Guardians of the Galaxy Vol. 2	-2017	7.6	587778
Star Wars: Episode VIII - The Last Jedi	-2017	7.0	576228
Dunkirk	-2017	7.8	573530
Spider-Man: Homecoming	-2017	7.4	531300
Get Out	(I) (2017)	7.7	508483
Blade Runner 2049	-2017	8.0	478688
It	(I) (2017)	7.3	470368
Baby Driver	-2017	7.6	455243
Three Billboards Outside Ebbing, Missouri	-2017	8.1	449900
Justice League	-2017	6.2	414431
Coco	(I) (2017)	8.4	405978
The Shape of Water	-2017	7.3	382087
John Wick: Chapter 2	-2017	7.5	378521
Jumanji: Welcome to the Jungle	-2017	6.9	329963
La casa de papel	(2017–)	8.3	323469
Dark	(2017–2020)	8.8	300932
Kong: Skull Island	-2017	6.6	288775
Kingsman: The Golden Circle	-2017	6.7	281422

The script for multiple pages will be covered in next class.

Thanks!