Hands-On Data Visualization

Interactive Storytelling from Spreadsheets to Code

Jack Dougherty Ilya Ilyankou

2020-05-26

Contents

Introduction

This open-access **book-in-progress**, by Jack Dougherty and Ilya Ilyankou, is under contract with O'Reilly Media, Inc., and was last updated on: 26 May 2020

Tell your story and show it with data, using free and easy-to-learn tools on the web. This introductory book teaches you how to design interactive charts and customized maps for your website, beginning with easy drag-and-drop tools, such as Google Sheets and Tableau Public, then gradually working up to editing open-source Chart.js and Leaflet code templates on GitHub. Follow along with the step-by-step tutorials, real-world examples, and online resources. This book is ideal for students, non-profit organizations, small business owners, local governments, journalists, academics, or anyone who wants to tell their story and show the data. No coding experience is required.

Read for free online at https://HandsOnDataViz.org or purchase print/eBook editions, to come from the publisher.

Please send corrections or suggestions for this book-in-progress to handsondataviz@gmail.com, or open an issue or submit a pull request on its GitHub repository. If you submit a GitHub pull request, in your commit message, please add the sentence "I assign the copyright of this contribution to Jack Dougherty and Ilya Ilyankou," so that we can maintain the option of publishing this book in other forms.

View open-source code for source text and templates at https://github.com/handsondataviz.

Trademarks

Any use of a trademarked name without a trademark symbol is for readability purposes only. We have no intention of infringing on the trademark.

- GitHub and the GitHub logo are registered trademarks of GitHub, Inc.
- Google and the Google logo are registered trademarks of Google Inc.
- WordPress is a registered trademark of the WordPress Foundation

Disclaimer

The information is this book is provided without warranty. The lead author, contributors, and publisher have neither liability nor responsibility to any person or entity related to any loss or damages arising from the information contained in this book.

Authors

Authors	About Us
	Jack Dougherty is Professor of Educational Studies at Trinity College in Hartford, Connecticut, where he and his students partner with community organizations to help tell their data stories on the web. Follow him on Twitter and on GitHub.
	Ilya Ilyankou is a Civic Technologist at Connecticut Data Collaborative. He has completed a double major in Computer Science and Studio Arts in the Class of 2018 at Trinity College. Visit his website or follow him on GitHub.

Acknowledgements

An earlier version of this book was titled *Data Visualization For All* and designed to accompany a free online edX course by the same name at Trinity College. Two co-instructors for this edX course contributed valuable ideas and co-created videos: Stacy Lam, Trinity Class of 2019, and David Tatem, Instructional Technologist at Trinity College. Veronica X. Armendariz, Trinity Class of 2016, also made valuable contributions to an earlier version of the book while she was a Teaching Assistant for the DataViz internship seminar.

Videos for the edX course were produced with Trinity College Information Technology staff and friends: Angie Wolf, Sean Donnelly, Ron Perkins, Samuel Oyebefun, Phil Duffy, and Christopher Brown.

Funding for student contributors to an earlier version of this book was generously provided by the Community Learning Initiative and Information Technology Services at Trinity College in Hartford, Connecticut.

What is Data Visualization?

Data visualization is broadly defined as a method of encoding quantitative, relational, or spatial information into images. Classic examples include Charles Menard's figurative map of Napoleon's defeat and retreat during the Russian campaign of 1812, and John Snow's dot map of cholera cases during the London epidemic of 1854.

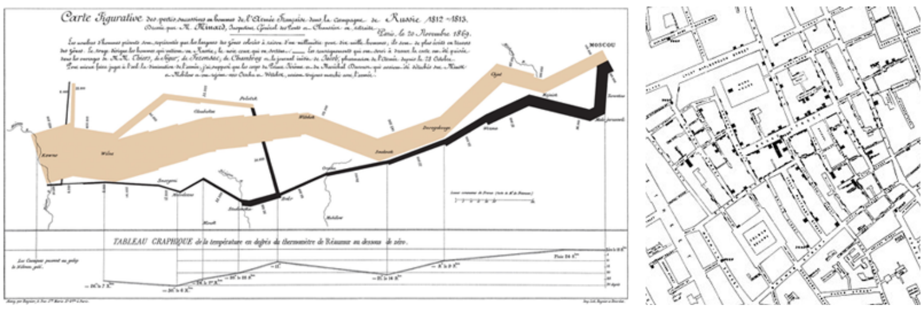


Figure 1: Images: Menard's figurative map (left) and Snow's dot map (right), from Wikimedia

This free online introductory book focuses on selected topics in data visualization:

Charts and maps Despite the growing variety of visualization types, this book features chapters on creating charts and maps, and a wide range of ways to communicate with these classic models.

Reusable tools and templates: Unlike infographics created for one-time use, all of the tools and templates in this book are recyclable, and allow you to upload a new dataset to display your story.

Free and easy-to-learn: We have selected data visualization tools that are free to use (or work on a freemium model, where advanced features or higher usage requires payment), and searched for those that we believe are easy-to-learn, based on our teaching experience with undergraduate students and non-profit community organizations.

Interactive on the open web: Many books assume that you will deliver your data visualizations to in-person audiences on printed paper or presentation slides. But in this book, we show how to embed interactive charts and maps on your website, to share with the wider public.

Storytelling: Data visualization is more than pretty pictures. In this book, the best visualizations are those that tell your data story – and pull readers' attention to what really matters – by combining images and text, and offering exploration with explanation.

Learn more

- Michael Friendly and Daniel J. Denis, "Milestones in the History of Thematic Cartography, Statistical Graphics, and Data Visualization," 2001, http://www.datavis.ca/milestones/
- Isabel Meirelles, Design for Information: An Introduction to the Histories, Theories, and Best Practices Behind Effective Information Visualizations (Rockport Publishers, 2013), http://isabelmeirelles.com/book-design-for-information/
- Edward Tufte, The Visual Display of Quantitative Information (Graphics Press, 1983), and subsequent works at https://www.edwardtufte.com

Why this book?

Hands-On Data Visualization, an open-access online textbook, seeks to help you tell your story—and show your data—through the power of the public web.

This open-access book reflects what I've learned while teaching data visualization to undergraduate students at Trinity College, and now to a global online class on the Trinity edX platform. Over the past few years, Trinity students and I have built interactive charts and maps in partnership with non-profit organizations in Hartford, Connecticut, to help them share their stories with data on the public web. Also, my students and colleagues have used these tools to create On The Line: How Schooling, Housing, and Civil Rights Shaped Hartford and its Suburbs, an open-access book-in-progress that features interactive historical maps of urban-suburban change. Students and colleagues who wrote tutorials, designed learning exercises, or developed code templates for *Hands-On Data Visualization* are listed as authors and contributors.

Although my outstanding colleagues have professional training, do not confuse them with me, the proverbial "Jack of all trades, master of none." I do not consider myself an expert in data visualization, nor should anyone mistake me for a computer scientist or data scientist. Inspect my higher education transcripts and you'll see only one computer science class (something called FORTRAN77 back in 1982), and not a single course in statistics, sadly. Instead, my desire to learn data visualization was driven by my need as an historian to tell stories about urban-suburban places and change over time. If you've ever watched me teach a class or deliver a presentation on these topics – always talking with my hands in the air – you'll understand my primal need to create charts and maps. Stories become more persuasive when supported with data, especially well-crafted images that convey data relationships more clearly than words. Furthermore, these data stories become more powerful when we share them online, where they reach broader audiences who can interact with and evaluate our evidence.

In the early 2000s, when I began to dabble in data visualization, our tools were expensive, not easy to learn, and not designed to share our stories on the public web. (One of my well-worn jokes is point to the bald spot on my head, and claim that it was caused while tearing out my hair in frustration while using ArcGIS.) But everything began to change around 2005 when Google Maps publicly released its application programming interface (API) that allowed people with some coding skills to show data points on an interactive web map. Gradually, between 2008-11, I began learning what was possible by working on map projects with talented programmers and geographers, such as Jean-Pierre Haeberly at Trinity, and Michael Howser at the University of Connecticut Libraries Map and Geographic Information Center (MAGIC, my favorite acronym), thanks to a grant from the National Endowment for the Humanities. Free and low-cost workshops sponsored by The Humanities and Technology Camp (THATCamp) at the Center for History and New Media at George Mason University, and Transparency Camp by the Sunlight Foundation, introduced me to many people (especially Mano Marks and Derek Eder) who demonstrated easier-to-use tools and templates, such as Google Fusion Tables and GitHub. Closer to home, Alvin Chang and other data journalists at the Connecticut Mirror showed me how to tell stories on the web with more flexible open-source tools, such as Leaflet and Highcharts.

All of these data visualization lessons I learned have been so valuable—to me, my students, our community partners, and thousands of readers on the web—that my co-authors and I have agreed to share our knowledge with everyone for free. This open-access book is guided by the principle of democratization of knowledge for the public good, hence the book's title: *Hands-On Data Visualization.* Not everyone can afford to make this choice, I realize. But the mission of Trinity College is to engage, connect, and transform, with both our local city of Hartford and the world at large. Since Trinity already pays my salary as a tenured professor, the right thing to do with the knowledge my students and I have gained is to pay it forward. That's why we created *Hands-On Data Visualization.*

If this free book is valuable for your education, then join us by sharing and supporting it for future readers:

- Tell your friends about the book and share the link via social media, text, or email
- Improve the book by adding comments or suggesting new chapters on our GitBook platform

Try out the tutorials, explore the online examples, share what you've learned with others, and dream about better ways to tell your data stories.

How to Read

This open-access book-in-progress is free to read online at http://HandsOnDataViz.org to fully experience the interactive charts, maps, and video clips. Any modern web browser will display the book, but readers may prefer larger screens (desktop, laptop, tablet) over smaller screens (such as smartphones). In your web browser, try these toolbar features near the top of the page:

- Menu
- Search
- Font to adjust text size and display
- View source code on GitHub
- Shortcuts (arrow keys to navigate; `s` to toggle sidebar; `f` to toggle search)
- Social Media
- Share

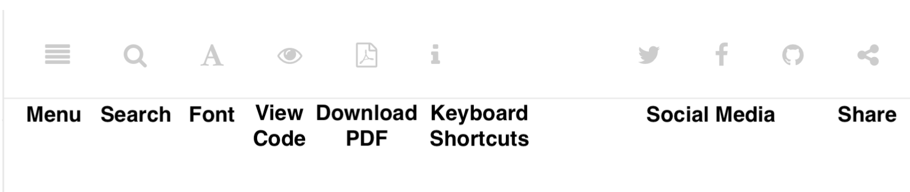


Figure 2: Screenshot: How to read

Open links in new tabs

Keep your place when reading online and moving between pages.

- Two-finger trackpad click
- or Control + click (Mac)
- or Alt + click (Chromebook)
- or right-click (Windows and others)

Use a second monitor

If you have a small screen, consider connecting a second monitor, or work next to a second computer or tablet. This allows you to view tutorials in one screen and build visualizations in the other screen.

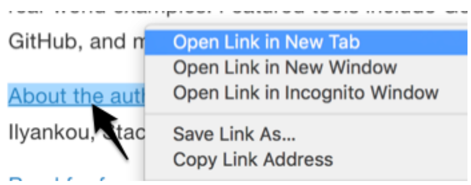


Figure 3: Screenshot: Open link in new tab (on Mac)

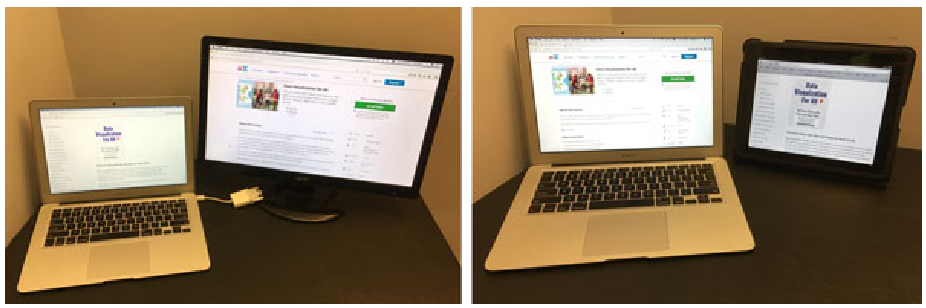


Figure 4: Image: Laptop with second monitor, and with tablet

Refresh browser

To view the most up-to-date content in your web browser, do a "hard refresh" to bypass any saved content in your browser cache.

- Ctrl + F5 (most Windows-Linux browsers)
- Command + Shift + R (Chrome or Firefox for Mac)
- Shift + Reload button toolbar (Safari for Mac)

Chapter 1

Choose Tools to Tell Your Data Story

Do you feel overwhelmed by the enormous range of data visualization tools? There's been so many different tools released in recent years that anyone would have a hard time deciding which ones to use. Even if you limit your choices to the dozen or so tools specifically mentioned in this book, how do you make wise decisions?

- Draw and Write Your Data Story reminds us to start with the most important item in your toolkit: ***your story***. Begin by drawing pictures and writing questions or sentences to capture your ideas on paper, and then choose the most appropriate tools to create your vision.
- Ask Questions When Choosing Tools lists several criteria to consider when making software decisions. Many of us look for free or affordable tools in the perfect sweet spot—easy-to-learn, yet powerful—and that's the focus of this book.
- Rate Three Simple Map Tools invites readers to create a basic interactive point map using three different online tools, and to evaluate each one using selected criteria from the chapter above.

Enroll in our free online course **TO DO add link**, which introduces these topics in the brief video below, and offers more exercises and opportunities to interact with instructors and other learners.

Watch the YouTube Video

Draw and Write Your Data Story

Before you dive deeply into software, think about the most important item in your toolkit: **your story**. The primary reason we're designing visualizations is to improve how we communicate our data story to other people, so let's begin there.

Push away the computer and pick up some old-school tools:

- colored markers or pencils
- lots of blank paper
- your imagination

First, at the top of the page, write down your data story.

- Is it in the form of a question? If so, figure out how to pose the question.
- Or maybe it's in the form of an answer to that question? If so, spell out your clearest statement.
- If you're lucky, perhaps you already can envision a full story, with a beginning, middle, and end.
- Whatever form it takes in your head, write out the words that come to mind.

Further down the page (or on a separate sheet), draw quick pictures of the visualizations that comes to your mind, even if you don't yet have any data. No artistic skills are required. Just use your imagination. - Do you envision some type of chart? Sketch a picture. - Or do you imagine some type of map? Show what it might look like. - Will your visualization be interactive? Insert arrows, buttons, whatever.

Finally, share your data story with someone else and talk through your preliminary ideas. Does your sketch and sentences help to convey the broader idea that you're trying to communicate? If so, this is one good sign that your data story is worth pursuing, with the visualization tools, templates, and techniques in other chapters of this book.

Ask Questions When Choosing Tools

When each of us decides which digital tools best fit our needs, we often face trade-offs. On one hand, many of us prefer easy-to-learn tools, especially those with a drag-and-drop interface, but they often force us to settle for limited options. On the other hand, we also favor powerful tools that allow us to control and customize our work, yet most of these require higher-level coding skills. The goal of this book is to find the best of both worlds: that "sweet spot" where tools are both friendly and flexible.

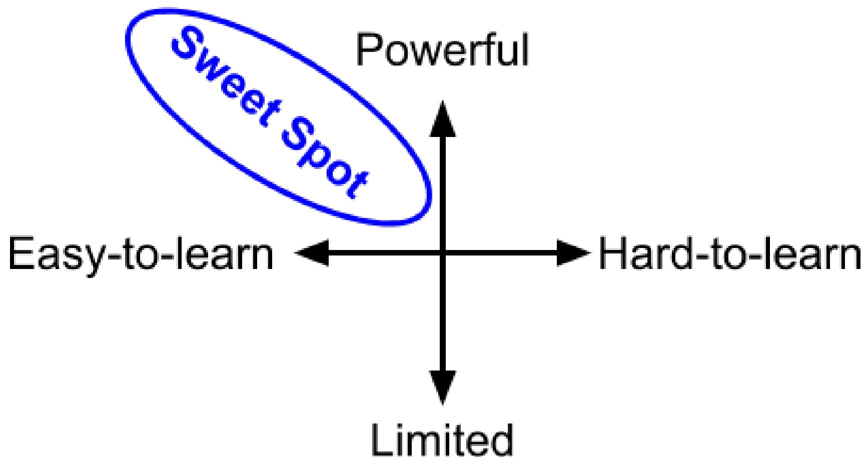


Figure 1.1: Diagram: the 'sweet spot' for easy-to-learn and powerful tools

Before testing out new tools, try listing the criteria that guide your decision-making process. What are the most important factors that influence whether or not you add another item to your digital toolkit? Here's the list that came to our minds:

1. Price: Is the tool free, or is there a "freemium" model to pay for more features or higher usage?
2. Easy-to-learn: Is the tool relatively simple for new users without coding skills?
3. Power: Does the tool support large amounts of data, and various types of visualizations?
4. Customization: Can I modify details about how my work appears?
5. Data Migration: Can I easily move my data in and out, in case I switch to a different tool? Hint for historians: Future-proof your digital history projects! Choose tools that allow you to easily export and migrate data to other platforms. Design projects to keep your data separate from its digital presentation.
6. Hosting: Can I decide exactly where my data and visualizations will be stored online?
7. Support: Is the tool actively maintained by its creators, and do they answer questions?

8. Open Source: Is the tool's software visible, can it be modified, and redistributed?
9. Security: Is the tool and my data protected from malicious hackers and malware?
10. Collaborative: Does the tool allow several people to work together on one shared product?
11. Privacy: Under the terms of service, is my data and work private or public?
12. Error-friendly: When something fails, does the tool point out possible problems and solutions?
13. Cross-platform: Does this tool work across different computer operating systems?
14. Mobile-friendly: Will it correctly display my work on various mobile devices and browsers?

That's a long list! It's even longer than the number of tools we'll mention in this book. But don't let it overwhelm you. The diagram at the top of the page illustrates the two most important criteria for the many free tools that are currently available: easy-to-learn and powerful features.

Learn more about choosing tools

Carl V. Lewis, Dataviz.tools: A curated guide to the best tools, resources and technologies for data visualization, http://dataviz.tools

Lincoln Mullen, "How to Make Prudent Choices About Your Tools," ProfHacker blog, Chronicle of Higher Education, August 14, 2013, http://www.chronicle.com/blogs/profhacker/how-to-make-prudent-choices-about-your-tools

Lisa Charlotte Rost, "What I Learned Recreating One Chart Using 24 Tools," Source, December 8, 2016, https://source.opennews.org/en-US/articles/what-i-learned-recreating-one-chart-using-24-tools/

Lisa Spiro and colleagues, DiRT: Digital Research Tools Directory (formerly Bamboo), http://dirtdirectory.org

Audrey Watters, " 'The Audrey Test': Or, What Should Every Techie Know About Education?," Hack Education, March 17, 2012, http://hackeducation.com/2012/03/17/what-every-techie-should-know-about-education

Rate Three Simple Map Tools

Let's explore criteria from the previous chapter by comparing three different tools, and reflecting on which factors you feel are most important when making decisions about your toolkit. We'll test three drag-and-drop tools to transform sample address data into a simple interactive point map.

Each tool can **geocode** address data by looking up a location (such as 500 Main Street, Hartford CT) in a large database, deciding on the best match, and converting this data into latitude and longitude coordinates (such as 41.762, -72.674).

For our sample data, we'll use this table of 9 locations in North America, with 3 intentional mistakes to test for geocoding errors.

Group	Description	Address
Government	US Capitol	East Capitol St NE & First St SE, Washington, DC 20004
Government	Parliament Hill	Wellington St, Ottawa, ON K1A 0A4, Canada
Government	Palacio Nacional	Plaza de la Constitución, Centro, 06000 Ciudad de México, CDMX, Mexico
Higher Education	Trinity College	300 Summit Street, Hartford, CT 06106
Higher Education	University of British Columbia	2329 West Mall, Vancouver, BC V6T 1Z4, Canada
Higher Education	Instituto Tecnológico de Monterrey	Av. Eugenio Garza Sada 2501 Sur, 64849 Monterrey, N.L., Mexico
Error Check	Incorrect spelling of Albuquerque	1801 Mountain Rd, Albakerky NM
Error Check	Imaginary street address	1 Stacylam Street, Chicago IL
Error Check	Street address without city	100 Main Street

Figure 1.2: Image: Sample address data screenshot

First, click this link and Save to download the sample file to your computer: sample-address-data in CSV format. CSV means comma-separated-values, a generic spreadsheet format that many tools can easily open. If you need help with downloading, see this short video tutorial.

Next, build a point map with the sample data, by following the tutorials for the three tools below.

Tool	Step-by-step tutorial in this book
Google My Maps	My Maps tutorial
Carto Builder	Carto tutorial

Finally, rate your experience using each tool with these selected criteria:

- Easy-to-learn: Which tool was the simplest for creating a basic point map?
- Price: Which of these free tools provided the most services at no cost?
- Customization: Which tool enabled you to modify the most details about your map?
- Data Migration: Which tool most easily allowed you to import and export your data?
- Error-friendly: Which tool geocoded most accurately or signaled possible errors?

Recommended: Enroll in our free online course **LINK TO DO** to compare your ratings to other students.

Chapter 2

Improve Your Spreadsheet Skills

Spreadsheets are wonderful tools to organize data into tables of rows and columns. With a spreadsheet, you can sort, filter, calculate, aggregate, and reorganize information to help you find the stories buried inside.

Four common spreadsheet tools:

Tool	Features
Google Sheets	Free, online collaborative spreadsheet on Google Drive. Requires free account.
Microsoft Excel Online	Free online spreadsheet on Microsoft OneDrive. Requires free account. Cannot open CSV generic spreadsheet files.
Microsoft Excel desktop	Not free (US $100+) spreadsheet for Mac or Windows desktop, as part of Microsoft Office.
LibreOffice	Free, open-source alternative to Microsoft Office desktop. Donation requested during download.

Which spreadsheet tool should you use? That depends on how you wish to share and store data for your project.

- If you are the **only person** working on a data project, use any spreadsheet tool.

- If you need to **protect private data**, avoid online tools and use any desktop spreadsheet.
- If you need to **share live data** with others, use Google Sheets.

This introductory online book features Google Sheets because it's a free and easy-to-learn tool for collaborating and sharing data with others. The basic spreadsheet methods shown here are very similar across all spreadsheet tools. But advanced users may need more complex tools to manage very large datasets, or relational databases, or to perform deeper analysis.

If you're new to spreadsheets or want to refresh your skills, see the following chapters:

- Upload and Convert to Google Sheets
- Make a Copy with Google Sheets
- Share with Google Sheets
- Save as CSV or ODS Format
- Sort and Filter Data
- Calculate with Formulas
- Group Data with Pivot Tables
- Match Data with VLookup
- Collect and Share Survey Data with Google Forms

Enroll in our free online course] which offers more spreadsheet exercises and opportunities to interact with instructors and other learners.

Upload Files and Convert to Google Sheets

Google Drive can convert many file types into Google Sheets format:

- Microsoft Excel (.xls and .xlsx)
- OpenDocument Spreadsheet (.ods)
- Comma-separated values (.csv)
- Tab-separated values (.tab)
- Text files (.txt) into Google Sheets format

Tutorial

1) Sign in to your free Google Drive account (http://drive.google.com)

2) To convert files into Google Sheets format, open the Settings (upper-right gear symbol), and **check the box** to Convert uploaded files to Google Docs.

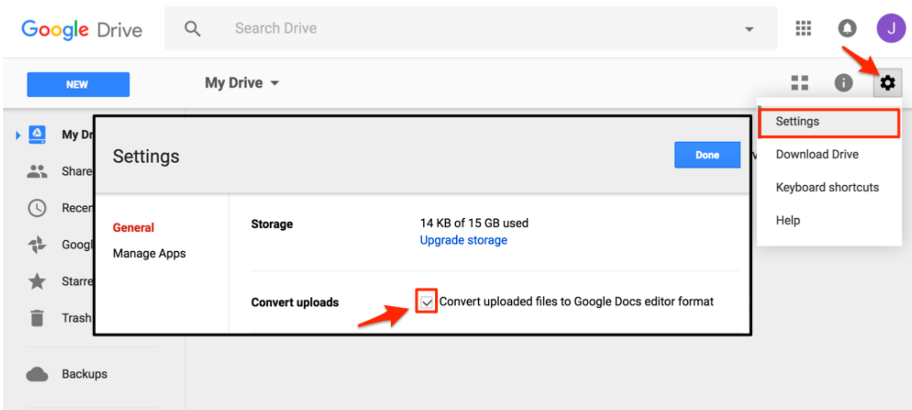


Figure 2.1: Screenshot: Check the box to Convert uploaded files to Google Docs format

3) To upload your file, use the New > File Upload menu OR drag-and-drop it into your Google Drive screen.

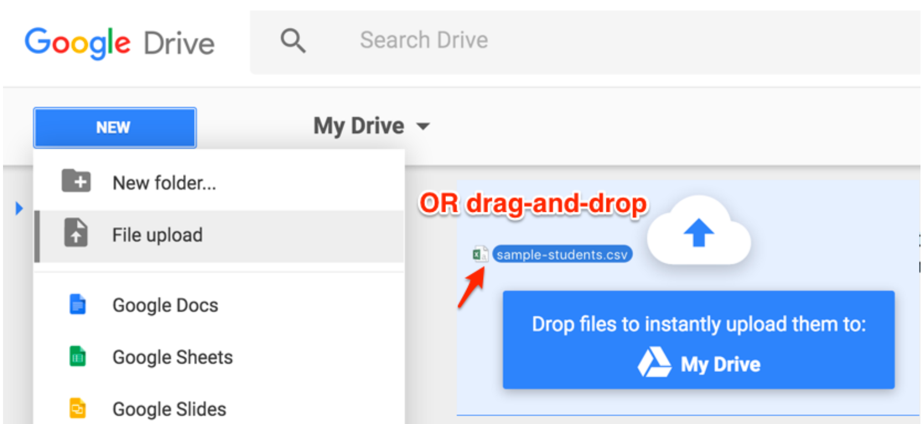


Figure 2.2: Screenshot: New > File upload OR Drag-and-drop the file

4) When your file is successfully converted, the Google Sheets icon will appear. Recommended: Right-click to rename the file and remove the old extension (.xlsx or .csv or other), since it is no longer in this old format.

5) Google Drive files that display different icons have **not** been converted into Google Sheets format.

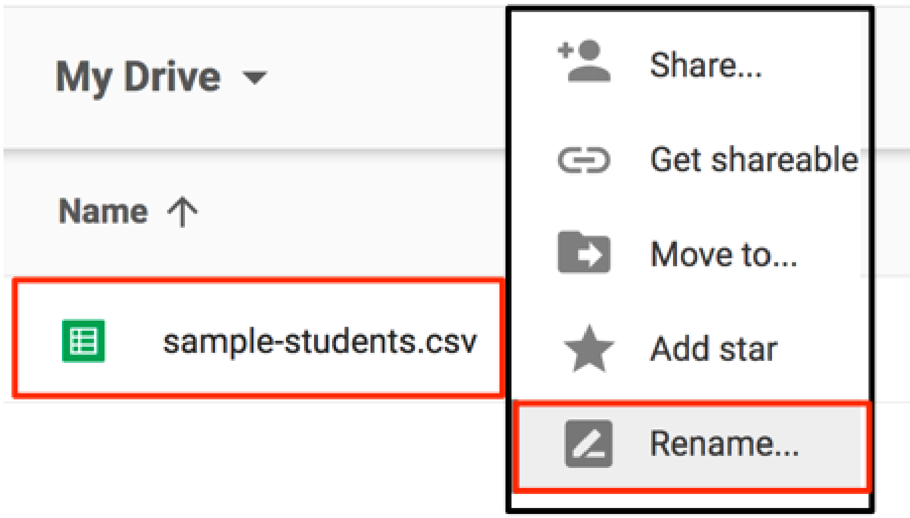


Figure 2.3: Screenshot: Right-click the Google Sheets icon to remove old file extension

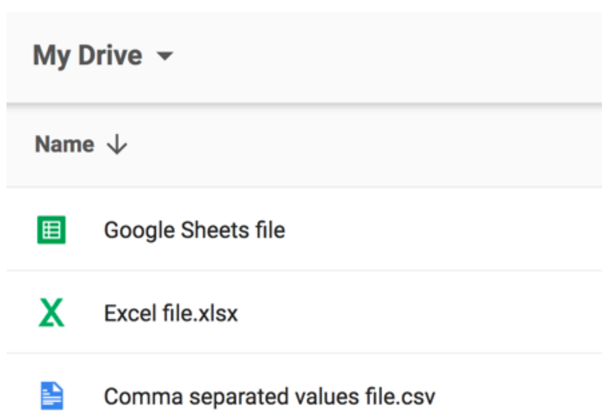


Figure 2.4: Screenshot: Spreadsheet format icons in Google Drive

Beware: A different way to convert spreadsheets into Google Sheets format is the File > Import menu, but this creates two files in your Google Drive (such as data and data.csv), which is confusing.

Make a Copy with Google Sheets

In this book, you will open links to Google Sheets that allow you to view – but not edit – the contents. How can you quickly make your own version that you can edit?

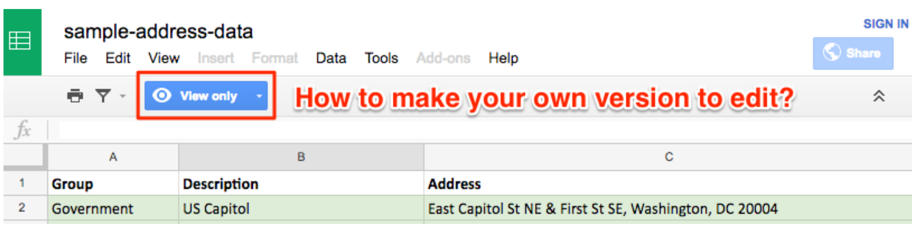


Figure 2.5: Screenshot: View-only in Google Sheets

Best solution

1) Sign in to your Google account in the upper-right corner. Requires a free account.
2) Go to File > Make a Copy to save a duplicate of the spreadsheet to your Google Drive. By default, your copy will be private to you. Go to the Share Data with Google Sheets chapter in this book to allow others to view, comment, or edit your spreadsheet.

Highly recommended: Create folders in your Google Drive to keep your files organized and easily findable.

Alternate solution

Another option is to File > Download As into a different format, such as:

- Microsoft Excel (.xlsx)
- OpenData System (.ods), a generic multi-tab spreadsheet
- Comma-separated values (.csv), a generic single sheet

No Google account is required.

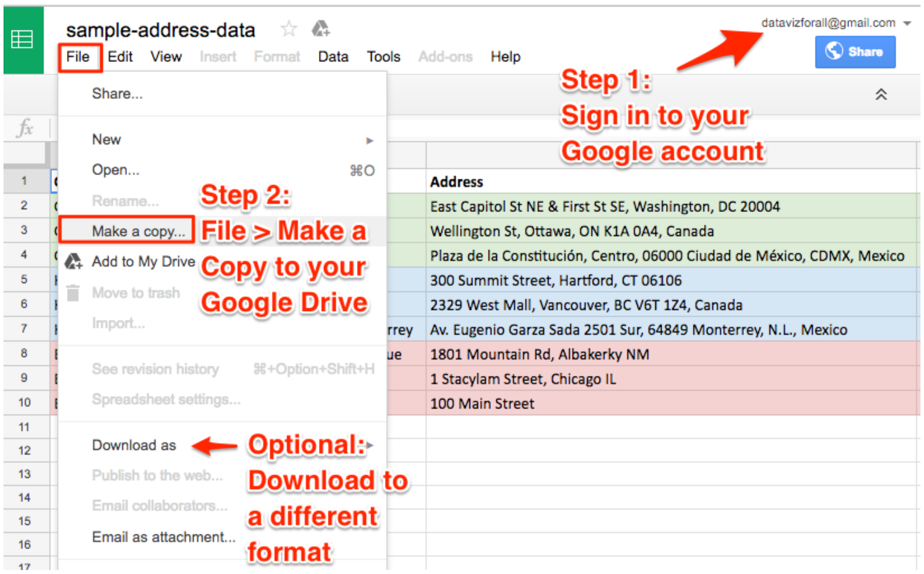


Figure 2.6: Screenshot: Sign in and File > Make a Copy in Google Sheets

Share Data with Google Sheets

To share live spreadsheet data with other people, use Google Sheets (https://www.google.com/sheets/about/). Requires a free Google Drive account.

Video with step-by-step tutorial

1) Sign in to your Google Drive (http://drive.google.com), and in the New menu, select Google Sheets.

2) New spreadsheets are private by default. Only the owner can view and edit.

3) To open your spreadsheet to others, click the blue Share button.

4) To share data with specific individuals, enter their Google usernames.

5) Or, to share data more widely, click the **Advanced** button on the next screen. (I wish Google made this button larger!)

6) Click the Change button and decide on Link Sharing settings:

- Public on the web (anyone can find your data)
- Anyone with the link (similar to an unlisted phone number)
- Off (only specific people you list by Google usernames)

Below those settings, select the Access level:

- Can view
- Can comment
- Can edit (for co-authored data)

7) Select Save, and scroll down on the next screen to select Done.

Tip: To avoid sending a long Google Sheets link to others, use a free link-shortening service such as Bit.ly (http://bit.ly). Requires a free account.

Learn more

- "Share Files from Google Drive," Google help page, https://support.google.com/docs/answer/2494822
- Jack Dougherty, "How to Co-Author and Peer Edit with Google Docs," Web Writing: How and Why for Liberal Arts Teaching and Learning, (2015), http://epress.trincoll.edu/webwriting/chapter/how-to-google-docs

Save Spreadsheets in CSV or ODS

To transfer spreadsheet data to another platform, or import it into a visualization tool, you may need to convert your file into a different format. Consider two options:

Comma-separated values (.csv)

- to transfer only one sheet of data, with no formulas or formatting, into a wide range of spreadsheet and visualization tools

OpenDocument Spreadsheet (.ods)

- to transfer multiple sheets, with basic formulas and formatting, into many spreadsheet tools (Excel, Google Sheets, LibreOffice)

Convert to CSV or ODS with Google Sheets

In the File > Download As menu, select either ODS (to convert a Google Sheets file with multiple tabs, formulas, and formatting) or CSV (to capture only the data in the current sheet).

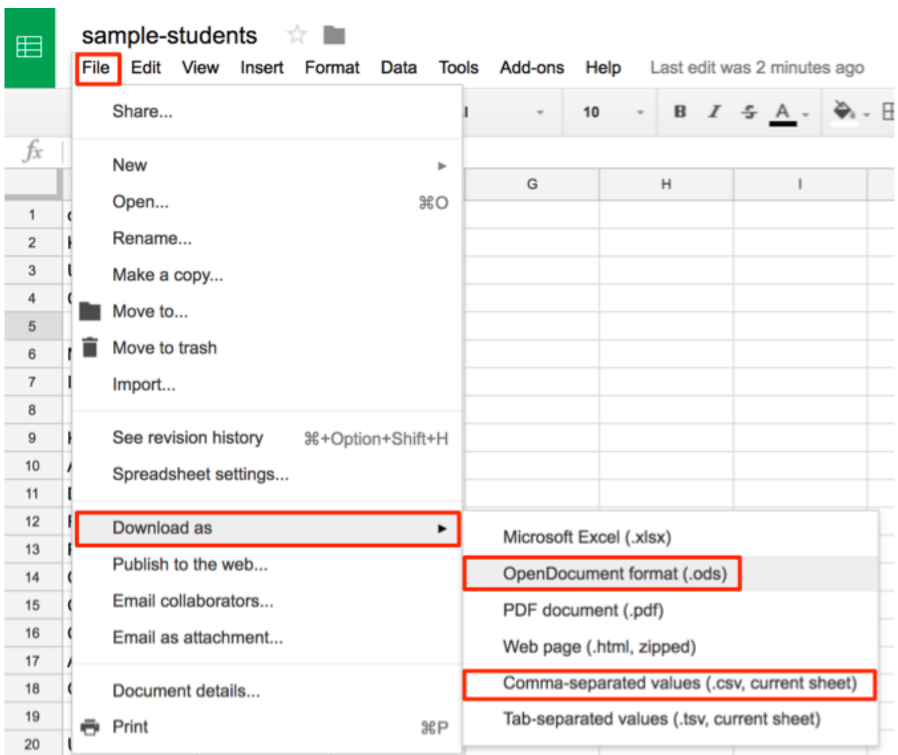


Figure 2.7: Screenshot: Download Google Sheets data in ODS or CSV format

Convert to ODS with Microsoft Excel

In the File > Save As menu, select ODS format.

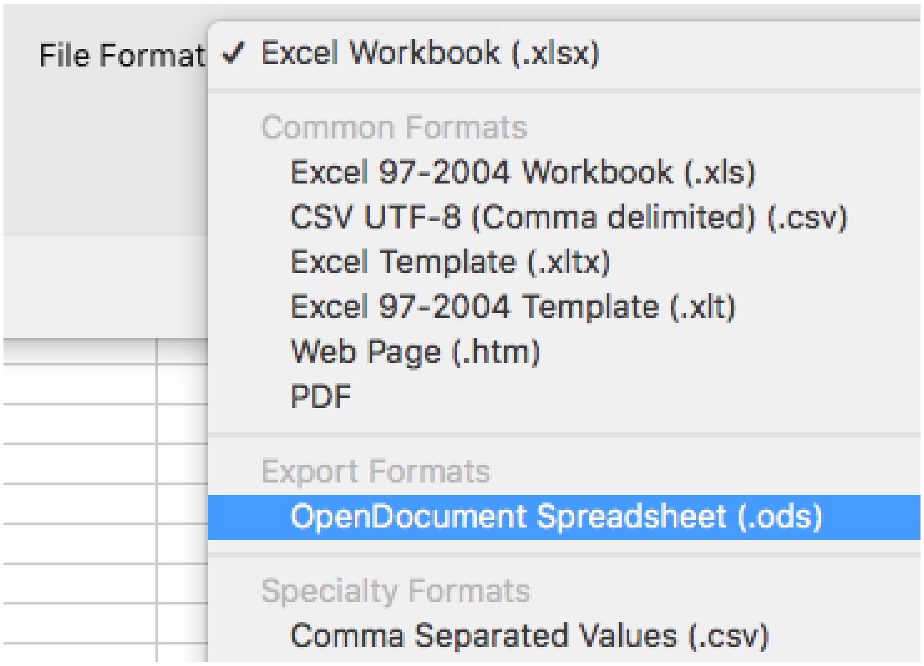


Figure 2.8: Screenshot: Save as ODS with Excel for Mac

Convert to CSV with Microsoft Excel

1) Note that CSV format will save only the first sheet of a multi-sheet Excel workbook. If you have source information or other data in other tabs, keep your original Excel file for backup purposes. You can give them parallel file names:

- data.csv
- data.xlsx

2) In the Excel file, select the File > Save As menu, and select CSV format.

3) Older versions of Excel may warn you that some features (such as formulas and formatting) will not be saved in a generic CSV data file. Be sure to keep a backup Excel version, then click Continue to save your data into CSV format.

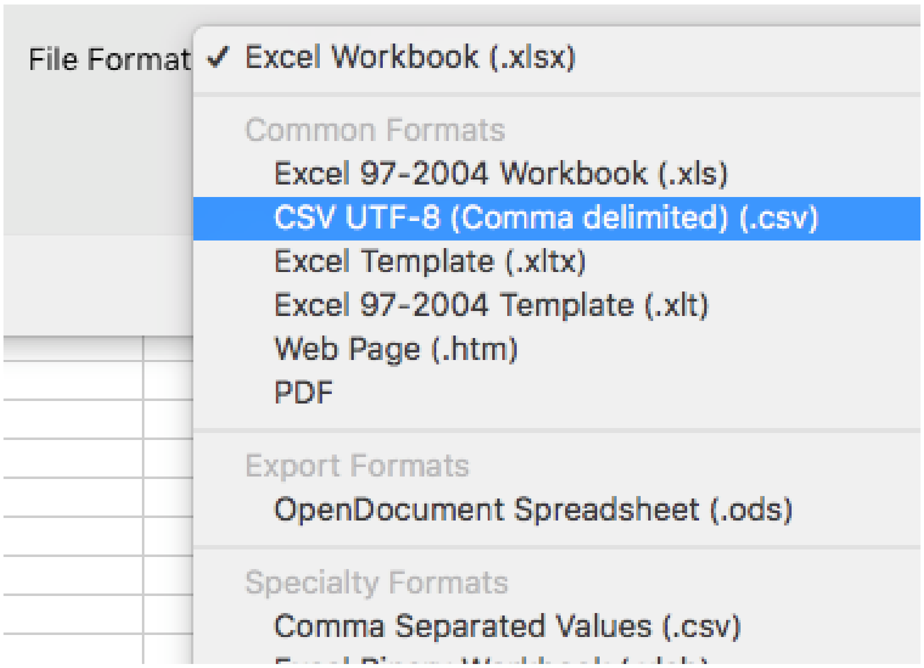


Figure 2.9: Screenshot: Save as CSV in Excel for Mac

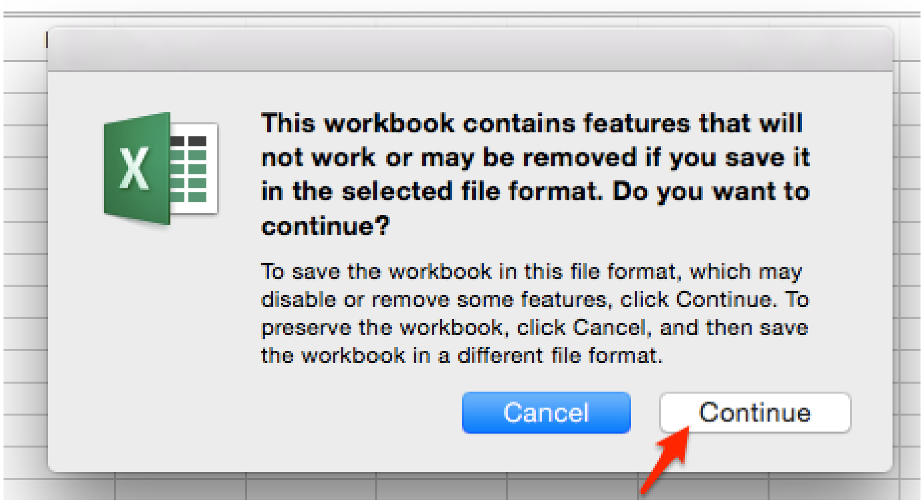


Figure 2.10: Screenshot: Older Excel warning before saving in CSV format

4) In older versions of Excel, when you quit the application, another screen will ask if you wish to save the CSV file a second time. **Don't let Excel confuse you.** If you have not made any changes to the Excel file since the step above, click Don't Save, because you already saved the file in CSV format.

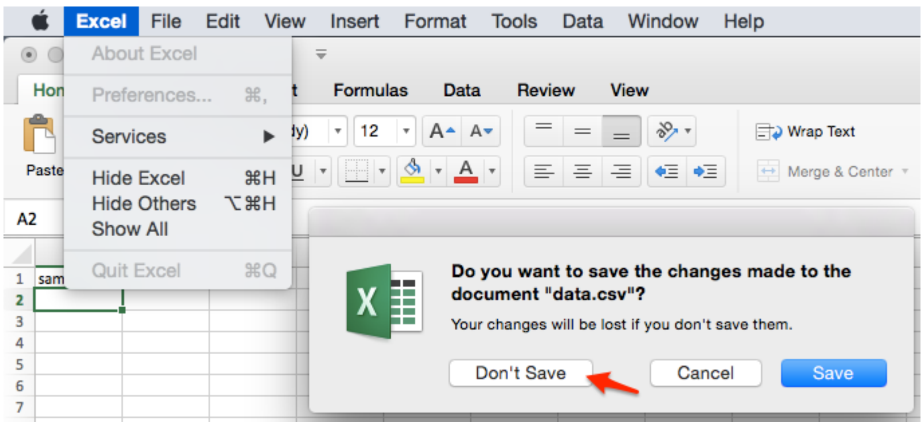


Figure 2.11: Screenshot: Older Excel version: click Don't Save

Sort and Filter Data

TODO:

- write intro on the title concepts
- when possible, start text by posing a common problem, and how this method can solve it
- redo visuals: Google Sheets with better example
- add Filter data

Sort data by columns

To sort data rows by a column, select the entire spreadsheet (top-left corner icon), then right-click or look for the sort menu. Be sure to select the entire sheet to avoid accidentally sorting one column without the adjacent ones.

Filter data by columns

TO DO

Calculate with Formulas and Functions

TODO:

- when possible, start text by posing a common problem, and how this method can solve it
- redo visuals: Google Sheets with better examples
- see other notes inserted below

Simple formulas can save you lots of time. The big advantage of spreadsheet tools is the ability to insert simple formulas to calculate numbers, or combine columns of text, for entire rows and columns.

Write a simple formula

In most spreadsheets, begin writing a simple formula with an equal sign, and refer to specific cells and functions, such as:

- = A2 + B2 + C2

Write formulas with built-in functions

TODO: rewrite to show how this is same as above

- = Sum(A2:C2)

TODO: rewrite to show common numerical and textual functions

- = Average(A2:C2)

Copy and paste, or drag formulas

If you've inserted a formula into one row, how can you quickly do the same calculation across all rows?

Spreadsheets can magically automate calculations across rows or columns. In most cases, you can copy and paste a formula into new cells. Sometimes you can click-and-drag the lower-right corner of a formula cell (which may appear as a cross-hair) to automate calculations.

Copy and Paste > Special > Values to replace formulas with data

After inserting calculations in a spreadsheet, sometimes dynamic formulas must be replaced with static data before the results can be visualized. One solution is to select and copy a column (or the entire sheet), then paste > special > values to replace the formula with numerical results.

Remember that if you need to check or run the calculations again at a later point, click (or right-click) the tab to save a copy to the spreadsheet as a backup.

Create a column of consecutive numbers

To quickly create a column of consecutive numbers, such as unique ID numbers, in most spreadsheet tools:

- Insert the number 1 into a cell and press Return
- Click the cell and float the cursor over the bottom-right corner, where it will change into a cross-hair symbol
- On a Mac, hold down the Option key and drag the cross-hair down to create consecutive numbers
- **TO DO** insert equivalent commands for Windows, Chromebook

Group Data with Pivot Tables

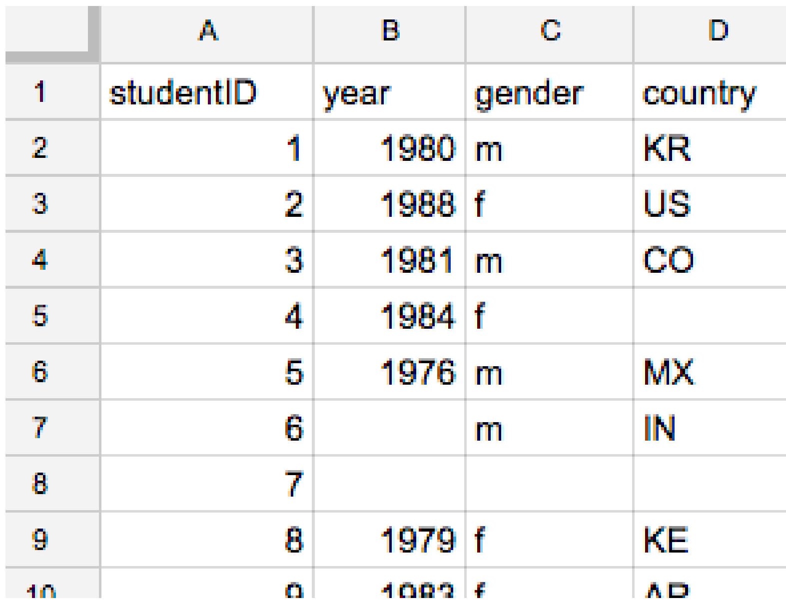
Here's a common problem: You open a large spreadsheet with many rows of data, such as a list of students. Your goal is to count students by categories, such as the number of students by each year of birth. What's the most efficient way to do this?

A solution: Create a pivot table to aggregate (or group together) and summarize data in another spreadsheet tab.

While pivot tables may look different across spreadsheet tools, the concept is the same.

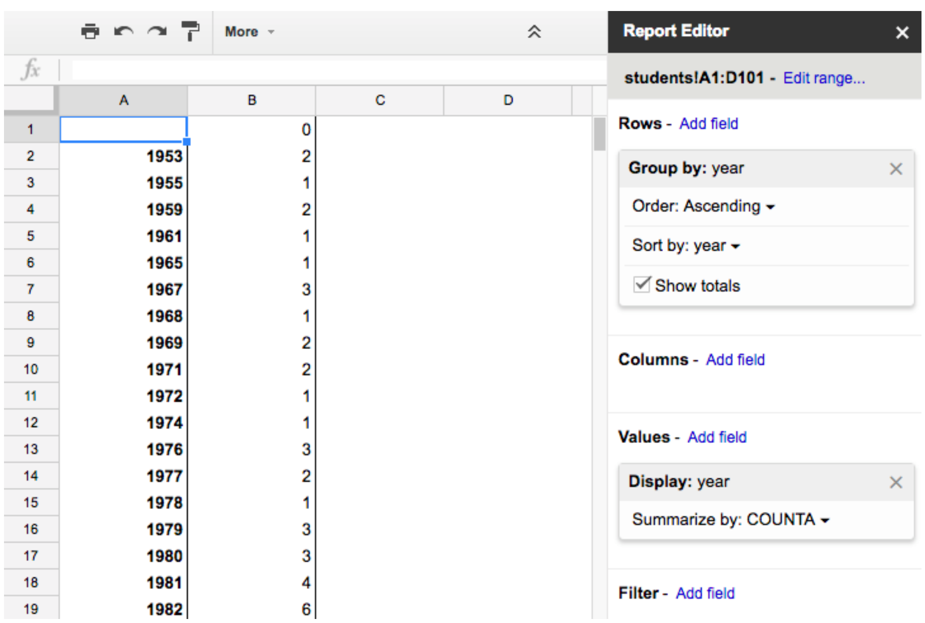
Video with step-by-step tutorial for Google Sheets

1) Click this link and Save to download to your computer: sample-students in CSV format. CSV means comma-separated values, a generic spreadsheet format that most tools can easily open.



	A	B	C	D
1	studentID	year	gender	country
2	1	1980	m	KR
3	2	1988	f	US
4	3	1981	m	CO
5	4	1984	f	
6	5	1976	m	MX
7	6		m	IN
8	7			
9	8	1979	f	KE

Figure 2.12: Screenshot: Long spreadsheet of student data



	A	B	C	D
1		0		
2	1953	2		
3	1955	1		
4	1959	2		
5	1961	1		
6	1965	1		
7	1967	3		
8	1968	1		
9	1969	2		
10	1971	2		
11	1972	1		
12	1974	1		
13	1976	3		
14	1977	2		
15	1978	1		
16	1979	3		
17	1980	3		
18	1981	4		
19	1982	6		

Figure 2.13: Screenshot: Pivot table of count by year of birth

2) Sign into Google Drive (requires free account) and drag-and-drop the sample CSV file to instantly upload. Before you do this, make sure your Settings (gear symbol) is set to Convert Uploads to Google Docs editor format (the default setting).

3) Shift-click to select all columns that you wish to pivot.

4) Select Data > Pivot Table…, which opens a new spreadsheet tab.

5) In Report Editor, select Rows > Add Field > Year to list all entries in order.

6) In Report Editor, select Values > Add Field > Year to summarize all values for each entry.

7) Change Summarize by SUM to Summarize by COUNTA (to count alphabetical or numerical entries), or COUNT (to count only numeric values).

More Advanced Pivot Table with Google Sheets

In addition to grouping by rows, you can create more advanced pivot tables by grouping by columns and filtering results. For example, the pivot table shown below shows rows by birth year, columns by gender (blank, female, male, other), and filters results to show only 18 students from one country: US.

Learn More

- Google, Create and Use Pivot Tables Help Page https://support.google.com/docs/answer/1272898
- LibreOffice, Creating Pivot Tables Help Page https://help.libreoffice.org/Calc/Creating_Pivot_Tables
- Andrew Ba Tran, "Tutorial: How to Make Pivot Tables in Google Sheets," TrendCT, September 4, 2015, http://trendct.org/2015/09/04/tutorial-how-to-make-pivot-tables-in-google-sheets

Match Columns with VLOOKUP

Here's a common problem: Sheet 1 contains a long roster of students enrolled in our *Data Visualization For All* course, with a two-letter code for their nation. Sheet 2 contains the list of codes for each nation. How can we quickly match up this information in one sheet, so that each row contains the nation for each student?

One solution: Spreadsheets contain a VLOOKUP function, which "looks up" data across two or more vertical columns, and automatically fills in matching

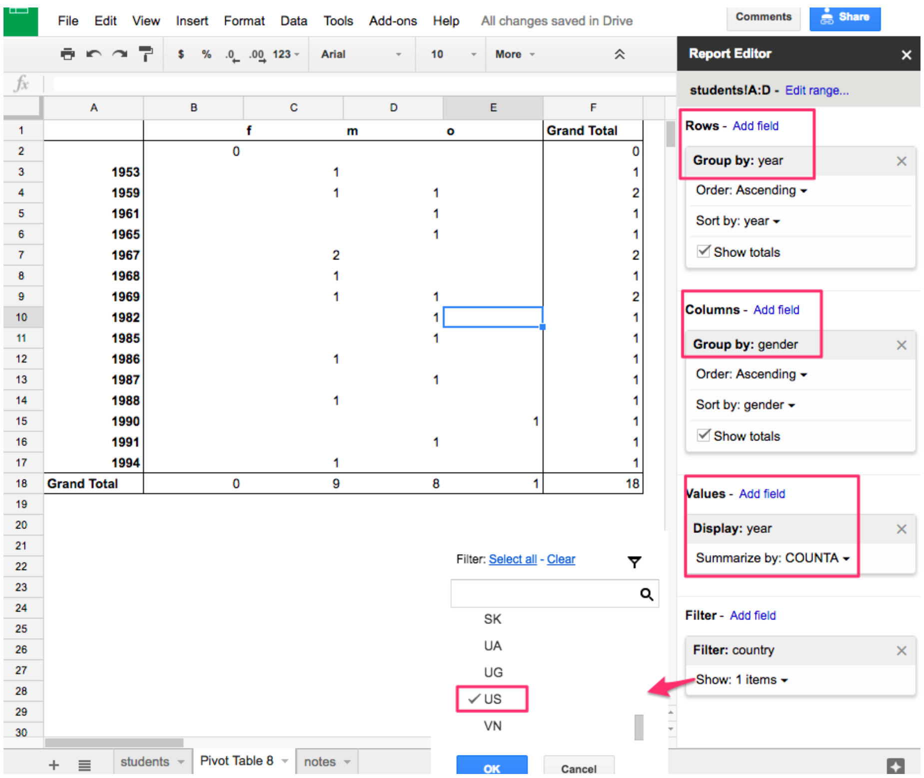


Figure 2.14: Screenshot: Advanced pivot table by year of birth and gender for US

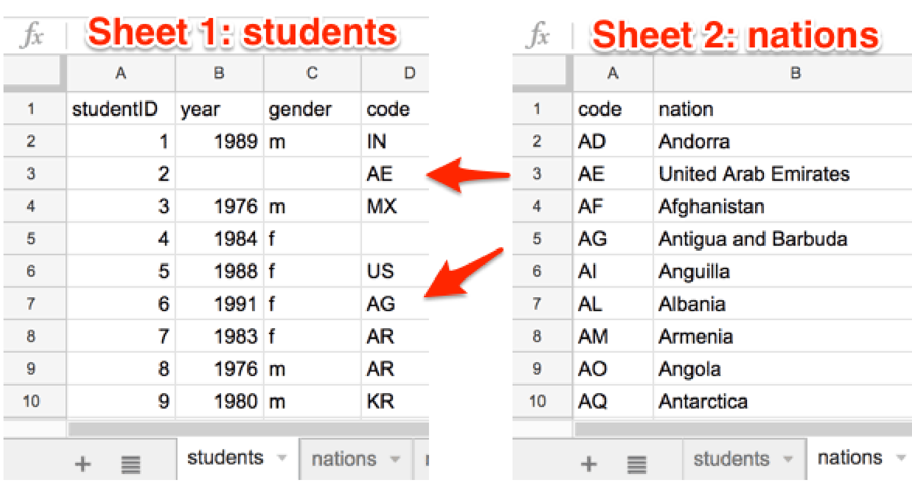
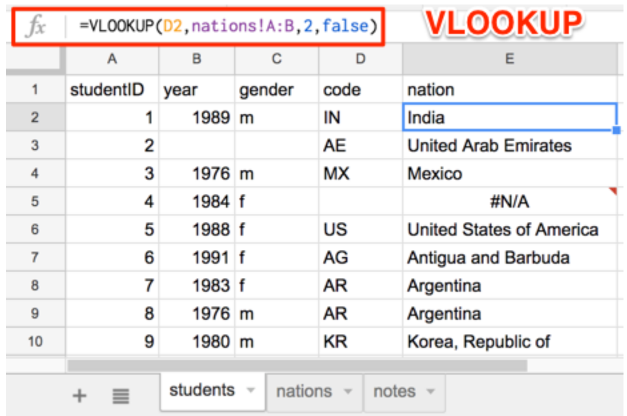


Figure 2.15: Screenshot: Problem - How to match columns in two sheets?

entries. This tutorial demonstrates how to set up this calculation in Google Sheets and Excel



	A	B	C	D	E
1	studentID	year	gender	code	nation
2	1	1989	m	IN	India
3	2			AE	United Arab Emirates
4	3	1976	m	MX	Mexico
5	4	1984	f		#N/A
6	5	1988	f	US	United States of America
7	6	1991	f	AG	Antigua and Barbuda
8	7	1983	f	AR	Argentina
9	8	1976	m	AR	Argentina
10	9	1980	m	KR	Korea, Republic of

Figure 2.16: Screenshot: Solution - Use the VLookup function

Video with step-by-step tutorial for Google Sheets

1) Click this link and Save to download to your computer: sample-students-nations in .ODS format. ODS means OpenDocument System, a generic multi-tab format that most spreadsheet tools can easily open.

2) To upload the downloaded file to Google Sheets, see the Upload Files and Convert tutorial in this book, and remember that Settings (gear symbol) must be set to Convert files to Google format. Or, open the file with Microsoft Excel or LibreOffice, and the directions below will be similar.

3) In the students sheet, type "nation" as a column header into cell E1.

4) Click in cell E2, start typing "=VLOOKUP" and the spreadsheet tool will suggest that you complete the formula in this format:

```
VLOOKUP(search_key, range, index, [is_sorted])
```

- search_key = the Sheet 1 cell we are trying to match
- range = the columns in Sheet 2 where matches may exist
- index = the column in the Sheet 2 range that contains the desired result, where 1 = first column, 2 = second column, etc.
- [is_sorted] = if the first column of the range is sorted, enter "true" to find the closest match; otherwise enter "false" to return exact matches only

4) You can type in the formula, or fill it out by clicking on cells, columns, and sheets as shown in the video above.

Collect and Share Data with Google Forms

TODO: write simple tutorial for Google Forms and explain how to share the spreadsheet; also mention other web form services

Chapter 3

Find and Know Your Data

Searching for open data: Increasing numbers of governmental agencies and non-profit organizations are publicly sharing *open data* on the web. When starting a new data visualization project, ask yourself these questions:

- Do I have the most relevant data for my project?
- Is it the most current data, in the most user-friendly format?
- Is data available at the individual level, or aggregated into larger groups?
- Which organizations might have collected data for my topic?
- Which open data repositories might have published this data?

What features do open repositories offer?

- View and export: At minimum, most open data repositories allow users to view their data and export it into common spreadsheet formats. Some also provide geographical boundaries for polygon maps.
- Built-in visualization tools: Some repositories offer built-in tools for users to create interactive charts or maps on the platform site. Some also provide code snippets for users to embed these built-in visualizations into their own websites.
- Static and Live data: Most repositories offer static datasets for a specific time period, but some also provide "live" data that is continuously updated.
- Application Programming Interface (APIs): Some repositories provide endpoints with code instructions that allow users to pull data directly from the platform into an external sites or online visualization, which is ideal for continuously updated data.

Know Your Data

Before starting to create charts or maps, get to know your data.

- Where did it come from?
- Who compiled the data, and for what purpose?
- What do the data labels really mean?
- Ask yourself: Am I working with the *most* recent version, in the *best* available format?

TODO: add resource https://github.com/Quartz/bad-data-guide

open data inception 1600+ sites portal http://opendatainception.io/

- Know your data: go out into the field to directly observe how the original data is measured and collected

https://www.opendatanetwork.com/

Closely examine your data files to understand their meaning, sources of origin, and limitations. TODO: expand on this theme with examples of bad and misleading data

1) Always ask: Am I using the best available data?

- Compare the HFS list to the City of Hartford's current list of food establishments: https://data.hartford.gov/browse
- go to Public Health Category
- click on the "dataset" version (updated 10 Feb 2016), which is same data but different view than the "map" version
- click on light blue "export" button into any format you wish to compare with the HFS list (see screenshot)
- decide which list is best for your organization's goal

US and Census Bureau Open Data

The U.S. Census Bureau collects and shares population, housing, and economic data on its open repositories.

- The Decennial Census is a full count of the population every ten years, most recently in 2010 and the upcoming one in 2020. Because decennial data are counts and not estimates, they represent "true" values and hence come without margins of errors.

- The American Community Survey (ACS) (https://www.census.gov/programs-surveys/acs/) is annual sample count, which produces:
  - 1-year estimates for areas with populations of 65,000+
  - 5-year estimates for all census areas
  - ACS used to release 3-year estimates for geographies with population of 20,000+, but discontinued after the 2011-2013 release.

Because ACS produces estimates and not "true" counts, data comes with margins of errors. Generally, margins of errors are higher for smaller geographies (eg census blocks) and smaller values (eg the number of Asian females aged 60+ who live in Union, CT). Hence, one needs to be critical when using ACS or other survey data.

Census areas are geographic divisions in this *general format*:

- State
- County
- County subdivisions (equivalent to Connecticut towns and cities)
- Census tracts (designated areas, roughly 2,500 to 8,000 people)
- Block groups (sub-unit of tract, roughly 600 to 3,000 people)
- Census blocks (sub-unit of block group, but not always a city block)

Census areas in the Hartford region

The interactive map below illustrates hierarchical relations among geographical census entities for the Hartford region, from state to census block level.

Learn more: Explore the standard hierarchy of US Census geographic entities and definitions (https://www2.census.gov/geo/pdfs/reference/geodiagram.pdf)

See also in this book: Geocode addresses with the US Census Geocoder

Data.census.gov

Data.census.gov (https://data.census.gov) is the main platform to access US Census data. It provides an easy search across census and survey tables. There is an interface to view tables for various years and geographies, and a download button to save data as CSV or PDF. It replaced American FactFinder (https://factfinder.census.gov) in July 2019.

Social Explorer

Social Explorer (https://www.socialexplorer.com/) is a popular tool to view and download census and related demographic data, past and present. The platform allows users to create data maps that may be exported as static images or presentation slides. Social Explorer requires subscription, but many academic institutions provide access.

TODO: create tutorial on how to cleanly download census data from Social Explorer and Census.gov to join with geography, especially census tract numbers

Data.gov

Data.gov (https://www.data.gov/) is the official open data repository for US federal government agencies, managed by the US General Services Administration, and powered by an open-source CKAN and WordPress platform.

National Center for Education Statistics

National Center for Education Statistics (NCES) (https://nces.ed.gov/) is the primary federal agency for collecting and reporting education data.

- Elementary/Secondary Information System (ELSi) (https://nces.ed.gov/ccd/elsi) - create custom tables and charts from the Common Core of Data (CCD) and Private School Survey.

Boundaries

- TODO
- link and source files and scale
- http://mapstarter.com/

Source Your Data Files

Source your data. Spell out exactly where it came from, so that someone other than you, several years in the future, could understand its origin.

Label the file name

Everyone has seen examples of bad file names:

- data.xls
- bldgdatalist.csv
- data77.xls

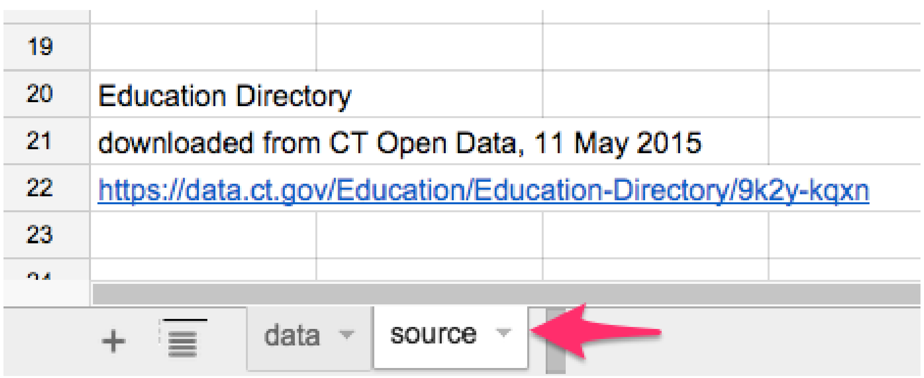
Write a short but meaningful file name. It is a good idea to include data source in file name (eg `acs2018`, `worldbank`, or `eurostat`). If different versions of the data are floating around, add the current date at the end, in YYYY-MM-DD format. Good file names look like this:

- town-demographics-2019-12-02.xls
- census2010_population_by_county.csv
- eurostat-1999-2019-CO2_emissions.xlsx

Save source data in separate sheet

Before modifying the original dataset, make sure to duplicate it to avoid any data losses. One way is to click (or right-click) on the spreadsheet tab to copy the sheet to another tab as a backup.

Add a *source* tab, after the data, with notes to remind you and others about its origins and when it was last updated.



Learn more

Lisa Charlotte Rost, *How to prepare your data for analysis and charting in Excel & Google Sheets*, https://blog.datawrapper.de/prepare-and-clean-up-data-for-data-visualization/

TODO: Source your data

```
- explain that data cannot be copyrighted, but representations of data can be
- open-source and creative commons
- credit sources and collaborators on dataviz products and readme files
- Whose perspectives does your data privilege? Whose stories remain untold?
```

Public or Private Data?

Many of the free web-based tools in this book require that your publicly share your data. Check each tool and decide whether it is appropriate for your data, which may have some privacy restrictions.

In some cases, individual data privacy is protected by law, but a government agency may aggregate (sort into larger groups) or anonymize (remove personally identifiable details) data to make it public. For example:

- Individual-level census data is private for about 70 years, but the US Census Bureau publicly releases anonymous data for aggregated areas (such as census blocks, tracts, towns, etc.)
- Patient-level health records are private, but public health officials share town- and county-level health data.
- Student-level education data is private, but school districts and state agencies publicly share grade-level and school-level data.

In other cases, individual data is not private. For example:

- When individuals contribute to political campaigns, most US and state laws require that the donor name, address, and amount is public data.
- When an individual buys home in Connecticut, the owner's name, address, purchase amount, and other details about the home are public data.

Know Your Data: Is It Good or Bad?

Before starting to create charts or maps, get to know your data.

- Where did it come from?
- Who compiled the data, and for what purpose?
- What do the data labels really mean?
- Ask yourself: Am I working with the *most* recent version, in the *best* available format?

Closely examine your data files to understand their meaning, sources of origin, and limitations. TODO: expand on this theme with examples of bad and misleading data

TODO: cite and explain this resource https://github.com/Quartz/bad-data-guide

Learn more

Christopher Ingraham, *An alarming number of scientific papers contain Excel errors*, https://www.washingtonpost.com/news/wonk/wp/2016/08/26/an-alarming-number-of-scientific-papers-contain-excel-errors/

Connecticut Open Data

Since this book was created in Hartford, Connecticut, we include state and municipal open data repositories and boundary files.

Connecticut Open Data (http://data.ct.gov), the official portal for state government agencies, is hosted on the Socrata platform, which offers built-in data visualization tools and APIs. See also how to create a filtered point map with Socrata in this book.

See also separate repositories for individual state agencies:

- Office of the State Comptroller (http://www.osc.ct.gov/openCT.html)
- CT State Department of Education (http://www.sde.ct.gov/sde/cwp/view.asp?a=2758&q=334520)
- Office of Policy and Management (http://ct.gov/opm/cwp/view.asp?a=3006&Q=383258&opmNav_GID=1386)
- link to all CT state government agencies (http://portal.ct.gov/Department-and-Agencies/)

Connecticut State Data Center (http://ctsdc.uconn.edu/), part of the U.S. Census Data Center Network, is the lead agency for US Census data and other socioeconomic data for Connecticut, and is based at the University of Connecticut Libraries. The site also features data visualizations created on the Tableau platform and provides population projections for the state of Connecticut.

MAGIC: The Map and Geographic Information Center (http://magic.lib.uconn.edu), based at the University of Connecticut Libraries, specializes in providing geographic, aerial photography, and map images for the state, past and present. The site also features interactive maps.

DataHaven (http://ctdatahaven.org/), a non-profit organization, collects and interprets information about Connecticut neighborhoods, such as its Community Wellbeing Survey. Data resources feature neighborhood profiles for densely-populated areas (New Haven and Hartford-West Hartford), and town profiles for other areas across the state.

Connecticut Data Collaborative (http://ctdata.org) is a public-private partnership that advocates for open data access to drive planning, policy, budgeting and decision making in Connecticut at the state, regional and local levels. We democratize public data through custom data exploration tools and a dynamic town profile tool, hosted on the open-source CKAN platform. Users can find state and federal data on topics such as public health, education, crime, municipal data, and racial profiling data.

Hartford Data (http://data.hartford.gov), the official portal of the City of Hartford municipal government, is hosted on the Socrata platform, which features built-in visualizations and APIs. See also how to create a filtered point map with Socrata in this book. Also, the Hartford Data site links to the City's ArcGIS Online geographic data (http://gisdata.hartford.gov/) and the City's financial data (http://checkbook.hartford.gov/) and budget (http://budget.hartford.gov/).

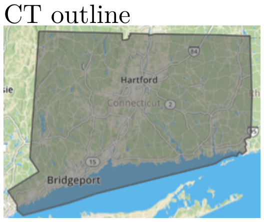
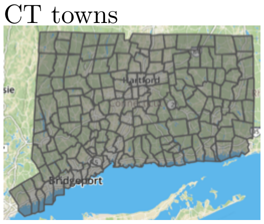
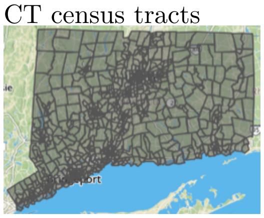
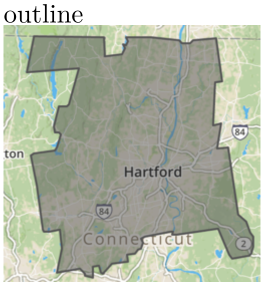
In addition to the official repositories above, Connecticut news organizations that create data visualizations often include links to download data files.

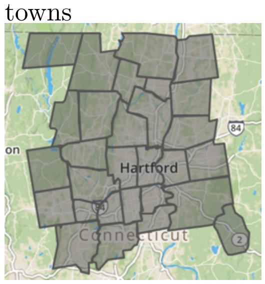
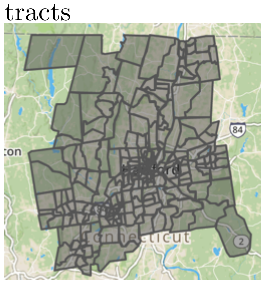
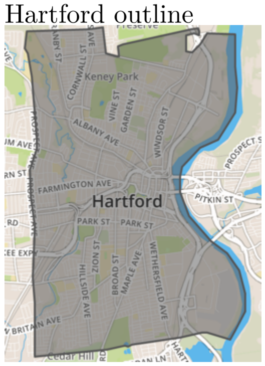
Connecticut Mirror / Trend CT (http://ctmirror.org/) and (http://trendct.org/) are publications of the Connecticut News Project, an independent, nonpartisan, nonprofit organization that focuses on state policy issues. Most of their data visualizations are built with open-source code, with publicly accessible data files. See also their GitHub repository (https://github.com/trendct).

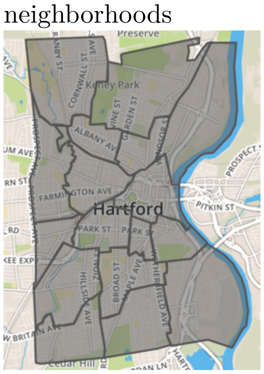
Hartford Courant Data Desk (http://www.courant.com/data-desk) produces digital visualizations for the *Hartford Courant*, the largest daily newspaper in Connecticut, owned by Tribune Publishing. Many of these data visualizations are published on the Tableau platform, which allows readers to download the underlying data.

Boundaries

- Converted from shapefile WGS84 to GeoJSON format
- To download a GeoJSON file, right-click the link and Save to your computer
- If you accidentally open the GeoJSON code in your browser, select File > Save Web Page to download it
- To view or edit, drag files into http://geojson.io or http://mapshaper.org
- Learn more in the Transform Your Map Data chapter of this book

Geography	Year-Source-Size	Right-click + Save to download GeoJSON
CT outline 	2010 Census UConn MAGIC WGS84 1:100,000	ct-outline.geojson
CT counties 	2010 Census UConn MAGIC WGS84 1:100,000	ct-counties.geojson
CT towns 	2010 Census UConn MAGIC WGS84 simplified to 224k	ct-towns.geojson
CT census tracts 	2010 Census UConn MAGIC WGS84 1:100,000	ct-tracts-2010.geojson
Hartford County outline 	2010 Census UConn MAGIC WGS84 1:100,000	hartfordcounty-outline.geojson

Geography	Year-Source-Size	Right-click + Save to download GeoJSON
Hartford County towns 	2010 Census UConn MAGIC WGS84 1:100,000	hartfordcounty-towns.geojson
Hartford County tracts 	2010 Census UConn MAGIC WGS84 1:100,000	hartfordcounty-tracts-2010.geojson
Hartford outline 	2010 Census UConn MAGIC WGS84 1:100,000	hartford-outline.geojson

Geography	Year-Source-Size	Right-click + Save to download GeoJSON
Hartford census tracts 	2010 Census UConn MAGIC WGS84 1:100,000	hartford-tracts-2010.geojson
Hartford neighborhoods 	2015 Hartford Open Data 1:50,000	hartford-neighborhoods.geojson

TODO: - add Capitol Region Council of Governments (CRCOG) http://www.crcog.org/ - add school districts (and clarify elementary-secondary) - add Capitol Region Education Council (CREC) http://www.crec.org/ - add school attendance areas from federal site - describe Freedom of Information Act (FOIA) data requests in Connecticut

Chapter 4

Clean Up Messy Data

TO DO

- write a new intro to match content that I moved into subfolders
- http://trendct.org/2015/08/28/getting-rid-of-duplicate-rows-using-google-sheets/
- Clean up data that contains stray commas, or mistyped entries
- Advanced clean up with Open Refine; see Alvin Chang's CT Mirror guide http://trendct.org/2015/04/24/john-jonathan-and-johnny-how-to-merge-them-in-open-refine/
- rethink formatting data
- see Jake Kara's "Data Structure Whining" https://github.com/jakekara/publishing-data-for-journalists

Clean Data with Spreadsheets

TODO: reorganize this to feature Google Sheets whenever possible, or Excel Online if needed

Sometimes we receive a spreadsheet with problematic data that needs to be cleaned up before we can successfully upload it into a visualization tool.

Find and Replace with a blank

A common problem with census data is that geographic names contain unnecessary words. For example, when downloading Connecticut county subdivisions (towns), each row appears as:

- Andover town

- Ansonia town
- Ashford town

Our goal is to remove the word “town” from each row, to produce a clean spreadsheet that we can match with other data, like this:

- Andover
- Ansonia
- Ashford

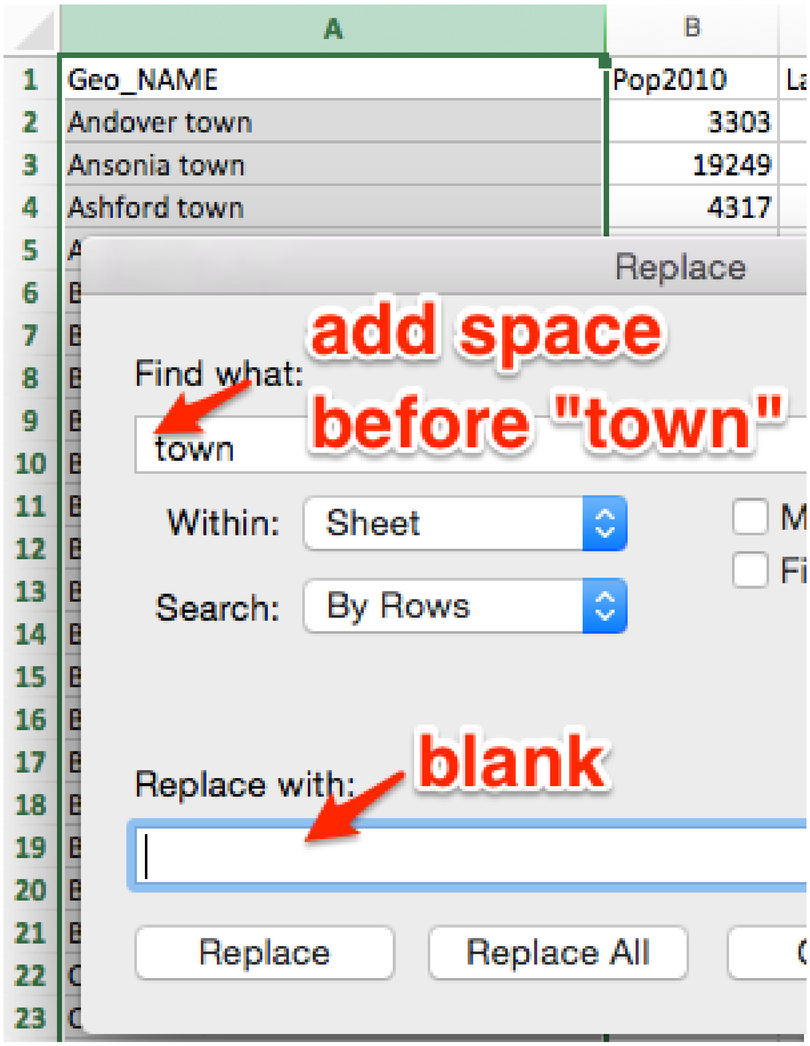
Here’s one quick solution: In any spreadsheet tool, use the Find and Replace command to remove unwanted characters. Try it! Click this link and Save to download to your computer:find-replace-town-geonames in CSV format. This tutorial shows screens from Excel, but other tools are very similar.

1. Open the Find and Replace command.

2. In the Find field, type " town“, leaving a space before the word, since we wish to remove only that word when by itself. (Otherwise, we would accidentally remove the”town" in Newtown.)

3. In the Replace field, leave it blank, to represent a blank space.

4. Press the Replace All button. Since this sample file lists 169 towns, the screen will states that 169 instances of “town” have been replaced.



Split one column into two with Excel

One common problem is when multiple pieces of data appear in one column, and your goal is to split them into separate columns. If those data pieces are separated by commas (or similar punctuation), you might be able to fix this with a simple spreadsheet command: split text into columns.

Try it! Click this link and Save to download to your computer: split-coordinate-pairs in CSV format, and open with Excel. (TODO: test with other spreadsheet tools)

1. Select the data column you wish to split.

2. Select Data > Split Text to Column
3. In the wizard screen, select Delimited data and click next.
4. In step 2 of the wizard screen, check the "comma" box, since this symbol divides the data column. Click next.
5. In step 3 of the wizard screen, accept the default General format, and Finish.

The coordinate pairs column is now split into two separate columns. Relabel the headers: longitude and latitude.

Animated example from Excel for Windows (thanks `@f3mlat`):

TODO: write directions to split a single address cell "300 Summit St, Hartford CT 06106" into separate columns for address, city, state, zip

Combine separate data columns into one

Another common data cleaning problem is when you receive address data in separate columns, like this:

Street	City	State	Zip
100 Main St	Hartford	CT	06106

But your data visualization tool requires you to combine all of this terms into one location column, like this:

Location
100 Main St, Hartford, CT 06106

One easy solution is to write a simple spreadsheet formula to combine (or concatenate) terms, using ampersands (&) as connectors, and quotation marks around blank spaces as separators. For example, if a spreadsheet contained four columns, *Address, City, State Zip* (A-D), then in column E insert a new header named *Location* and a formula in this format:

- =A2 &" " & B2 &" " &C2 &" " &D2

	A	B	C	D	E
1	Address	City	State	Zip	Location
2	950 Main St	Hartford	CT	06103	950 Main St Hartford CT 06103
3	300 Summit St	Hartford	CT	06106	=A3&" "&B3&" "&C3&" "&D3

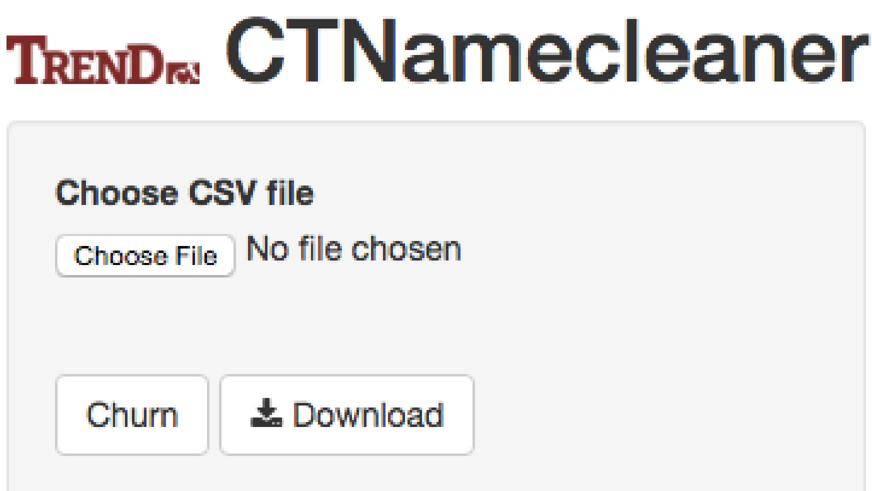
TODO:

- Confirm that Google Fusion Tables geocoder does not require commas between terms
- Clarify what happens with zip code in the example above

Convert Connecticut town names with CTNamecleaner

In Connecticut, residents often list their village or neighborhood names in their address, but these do not necessarily match the official list of 169 Connecticut town governments (called county subdivisions by the US Census). For example, the Elmwood neighborhood is located in the town of West Hartford, and the Rockville village is located in the town of Vernon.

To solve this problem, the data experts at TrendCT/CT Mirror have openly shared a wonderful tool to convert village/neighborhood names into official towns, called CTNamecleaner.



1. Open CTNamecleaner with your browser at http://shiny.trendct.org/ctnamecleaner/
2. Upload a CSV generic spreadsheet. Learn more about CSV format in this book **TO DO add link**.
3. Select the data column to be converted into town names, and download the results.

Learn more about CTNamecleaner on GitHub, and view the underlying list of Connecticut place names in a public Google sheet.

Clean Data with Open Refine

TODO: show basic tutorial with Open Refine; link to Alvin Chang's fabulous Open Refine tutorial in CT Mirror

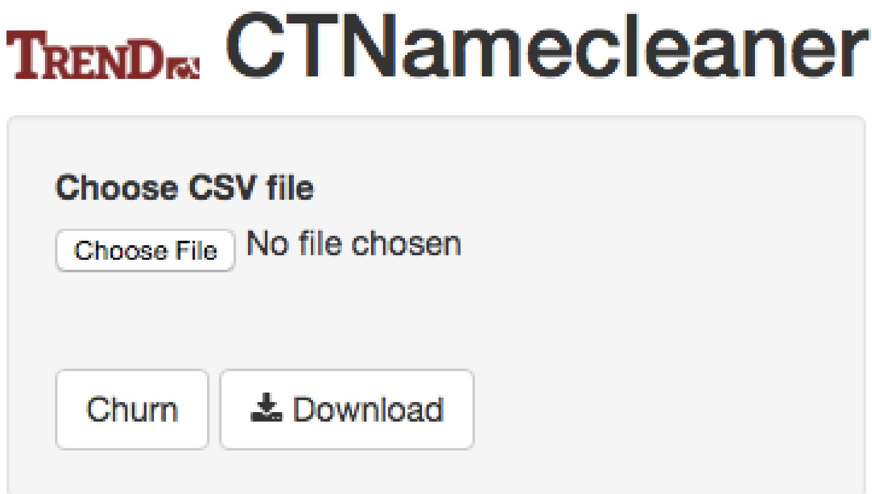
Fix Connecticut Town Names with CTNamecleaner

TODO: update this page; avoid duplication in main chapter text

Here's a wonderful data-cleaning tool that's specific to Connecticut, but the idea (and open-source code from TrendCT/CT Mirror) may inspire others to create similar tools for other locations.

In Connecticut, residents often list their village or neighborhood names in their address, but these do not necessarily match the official list of 169 Connecticut town governments (called county subdivisions by the US Census). For example, the Elmwood neighborhood is located in the town of West Hartford, and the Rockville village is located in the town of Vernon.

To solve this problem, the data experts at TrendCT/CT Mirror have openly shared a wonderful tool to convert village/neighborhood names into official towns, called CTNamecleaner.



1. Open CTNamecleaner with your browser at http://shiny.trendct.org/ctnamecleaner/
2. Upload a CSV generic spreadsheet. Learn more about CSV format in this book **TO DO** fix link
3. Select the data column to be converted into town names, and download the results.

Learn more about CTNamecleaner on GitHub, and view the underlying list of Connecticut place names in a public Google sheet.

Chapter 5

Chart Your Data

Charts pull readers deeper into your story. Even if your data contains geographical information, sometimes a chart tells your story better than a map. But designing meaningful, interactive charts requires careful thought about how to communicate your data story with your audience.

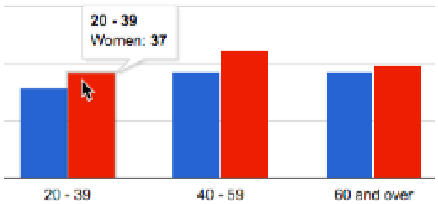
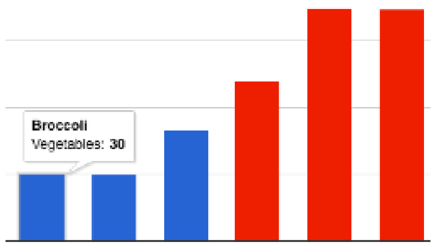
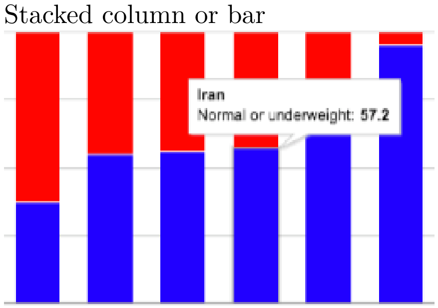
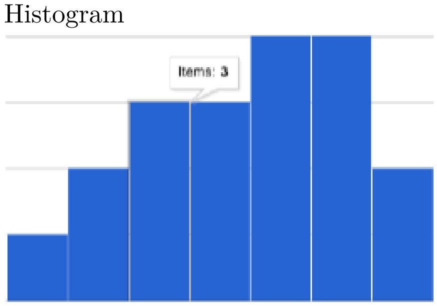
In this chapter, we will look at main principles of chart design, and learn to identify good charts from bad ones. You will learn to choose a chart type that matches your story and data format.

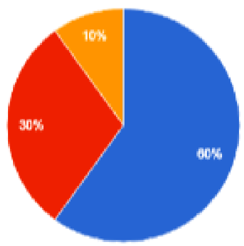
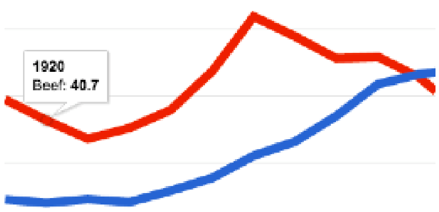
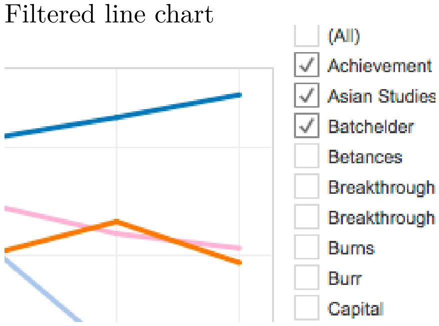
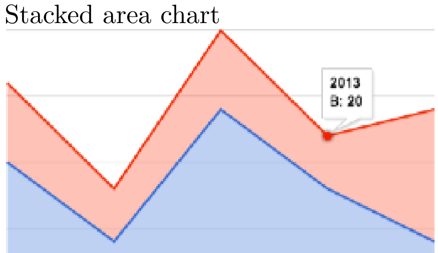
You will learn how to make static and interactive charts with Google Sheets and how to publish them on your website. We will then look at building interactive charts with Tableau Public, a free version of the powerful software used by data analysts and data visualization practitioners.

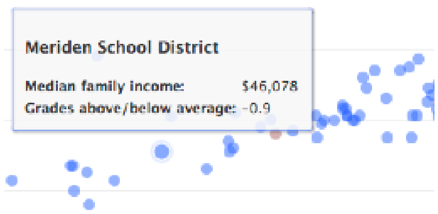
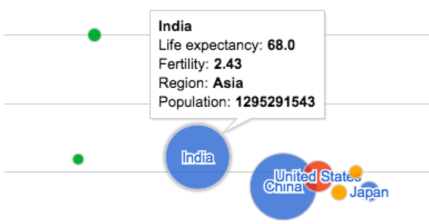
At the end, we will introduce chart templates with JavaScript's Chart.js library, which give you a lot of control over how the charts look. Working with Chart.js will require you to modify and host code templates with GitHub, which is described in detail in Chapter 8.

See also related chapters in this book:

- Draw and write your data story to capture your ideas on paper
- Improve spreadsheet skills, Find and know your data, and Clean your data
- Embed your interactive chart on your website
- Detect bias in data stories, including How to lie with charts
- Tell your data story, including its most meaningful insights and limitations

Basic chart types	Best use and tutorial chapters
Grouped column or bar 	Best to compare categories side-by-side. Vertical columns, or horizontal bars for long labels. Easy tool: Google Sheets bar and column tutorialPower tool: Chart.js templates
Separated column or bar 	Best to compare categories in separate clusters. Vertical columns, or horizontal bars for long labels.Easy tool: Google Sheets bar and column tutorialPower tool: Chart.js templates
Stacked column or bar 	Best to compare sub-categories, or parts of a whole. Vertical columns, or horizontal bars for long labels.Easy tool: Google Sheets bar and column tutorialPower tool: Chart.js templates
Histogram 	Best to show distribution of raw data, with number of values in each bucket.Easy tool: Google Sheets bar and column tutorialPower tool: Chart.js templates

Basic chart types	Best use and tutorial chapters
Pie chart 	Best to show parts of a whole, but hard to estimate size of slices.Easy tool: Google Sheets pie chart tutorialPower tool: Chart.js templates
Line chart 	Best to show continuous data, such as change over time.Easy tool: Google Sheets line chart tutorialPower tool: Chart.js templates
Filtered line chart 	Best to show multiple lines of continuous data, with on-off toggle buttons. Easy tool: Tableau Public filtered line chart tutorial
Stacked area chart 	Best to show parts of a whole, with change over time. Easy tool: Google Sheets stacked area tutorialPower tool: Chart.js templates

Basic chart types	Best use and tutorial chapters
Scatter chart 	Best to show relationship between two sets of data. Also called an XY chart. Easy tool: Google Sheets scatter chart tutorial or Tableau Public scatter chart tutorialPower tool: Chart.js templates
Bubble chart 	Best to show relationship between three or four sets of data, using bubble size and color.Easy tool: Google Sheets bubble chart tutorialPower tool: Chart.js templates

For more advanced chart types and tutorials

- Google Sheets Chart types help page
- Tableau Public resources page
- Chart.js samples page

Chart Design Principles

Although not a science, data visualization comes with a set of rules, principles, and best practices that create a basis for clear and eloquent charts. Some of those rules are less rigid than others, but prior to "breaking" them, it is important to establish why they are important.

Before you begin, ask yourself: Do I really need a chart to tell this data story? Or would a table or text alone do a better job? Making a good chart takes time and effort, so make sure it enhances your story.

5.0.1 Deconstructing a Chart

Let's take a look at Figure 5-1 that shows the basic components shared among most chart types.

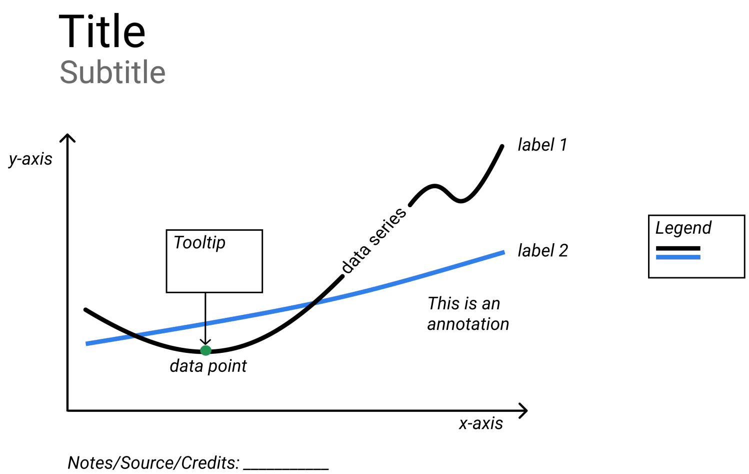


Figure 5.1: Figure 5-1

A *title* is perhaps the most important element of any chart. A good title is short, clear, and tells a story on its own. For example, "Black and Asian Population More Likely to Die of Covid-19", or "Millions of Tons of Plastic Enter the Ocean Every Year" are both good titles.

Sometimes a more "dry" and "technical" title is preferred. Our two titles can then be changed to "Covid-19 Deaths by Race in New York City, March 2020" and "Tons of Plastic Entering the Ocean, 1950–2020", respectively.

Often these two styles are combined into a title ("story") and a subtitle ("technical"), like that:

```
**Black and Asian Population More Likely to Die of Covid-19**
Covid-19 Deaths by Race in New York City, March 2020
```

Make sure your subtitle is less prominent than the title. You can achieve this by decreasing font size, or changing font color (or both).

Horizontal (x) and vertical (y) *axes* define the scale and units of measure.

A *data series* is a collection of observations, which is usually a row or a column of numbers, or *data points*, in your dataset.

Labels and *annotations* are often used across the chart to give more context. For example, a line chart showing US unemployment levels between 1900 and 2020 can have a "Great Depression" annotation arround 1930s, and "Covid-19 Impact" annotation for 2020, both representing spikes in unemployment. You might also choose to label items directly instead of relying on axes, which is common with bar charts. In that case, a relevant axis can be hidden and the chart will look less cluttered.

A *legend* shows symbology, such as colors and shapes used in the chart, and their meaning (usually values that they represent).

You should add any *Notes*, *Data Sources*, and *Credits* underneath the chart to give more context about where the data came from, how it was processed and analyzed, and who created the visualization. Remember that being open about these things helps build credibility and accountability.

In interactive charts, a *tooltip* is often used to provide more data or context once a user clicks or hovers over a data point or a data series. Tooltips are great for complex visualizations with multiple layers of data, because they de-clutter the chart. But because tooltips are harder to interact with on smaller screens, such as phones and tablets, and are invisible when the chart is printed, only rely on them to convey additional, nice-to-have information. Make sure all essential information is visible without any user interaction.

The Unbendable Rules

Although the vast majority of rules in data visualization are open to interpretation, there are some that are hard to bend.

Bar charts start with zero. Unlike line charts, bar or column charts need to have their value axis start at zero. This is to ensure that a bar twice the length of another bar represents twice its value. The Figure 5-2 shows a good and a bad example.

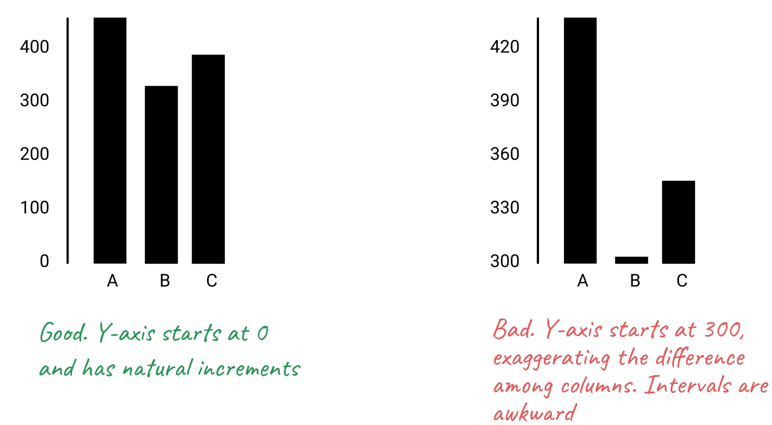


Figure 5.2: Figure 5-2

Starting y-axis at anything other than zero is a common trick used by some media and politicians to exaggerate differences in surveys and election results.

Pie Charts Represent 100%. Pie charts is one of the most contentious issues in data visualization. Most dataviz practitioners will recommend avoiding them entirely, saying that people are bad at accurately estimating sizes of different slices. We take a less dramatic stance, as long as you adhere to the recommendations we give in the next section.

But the one and only thing in data visualization that every single professional will agree on is that *pie charts represent 100% of the quantity*. If slices sum up to anything other than 100%, it is a crime. If you design a survey titled *Are you a cat or a dog person?* and include *I am both* as the third option, forget about putting the results into a pie chart.

Chart Aesthetics

Remember that you create a chart to help the reader understand the story, not to confuse them. Decide if you want to show absolute numbers, percentages, or percent changes, and do the math for your readers.

Start with a white background and add elements as you see appropriate. You should be able to justify each element you add. To do so, ask yorself: Does this element improve the chart, or can I drop it without decreasing readability? This way you won't end up with so-called "chart junk" as shown in Figure 5-3, which includes 3D perspectives, shadows, and unnecessary elements. They might have looked cool in early versions of Microsoft Office, but let's stay away from them today.

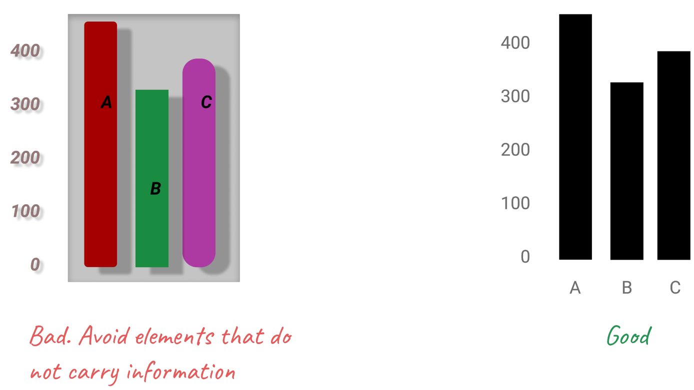


Figure 5.3: Figure 5-3

The only justification for using three dimensions is to plot three-dimensional data, which has x, y, and z values. And don't let anyone tell you otherwise.

Remember that pie charts only show part-to-whole relationship, so all slices need to add up to 100%. Generally, the fewer slices the better. It is a good practice to arrange slices from largest to smallest, clockwise, and put the largest slice at 12 o'clock. Figure 5-4a illustrates that.

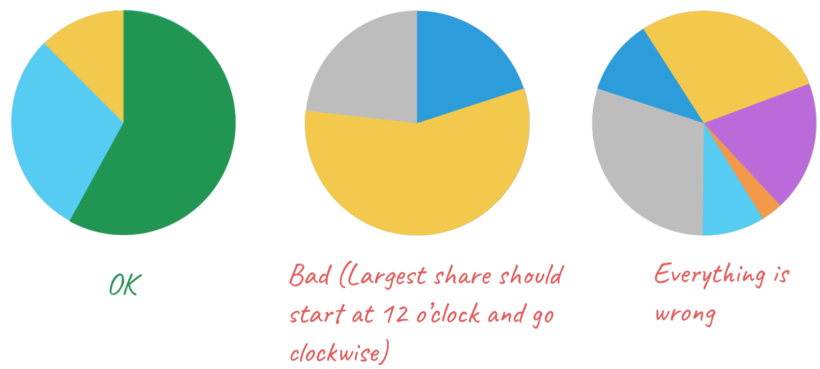


Figure 5.4: Figure 5-4a

If your pie chart has more than five slices, consider showing your data in a bar chart, either stacked or separated, like Figure 5-4 shows.

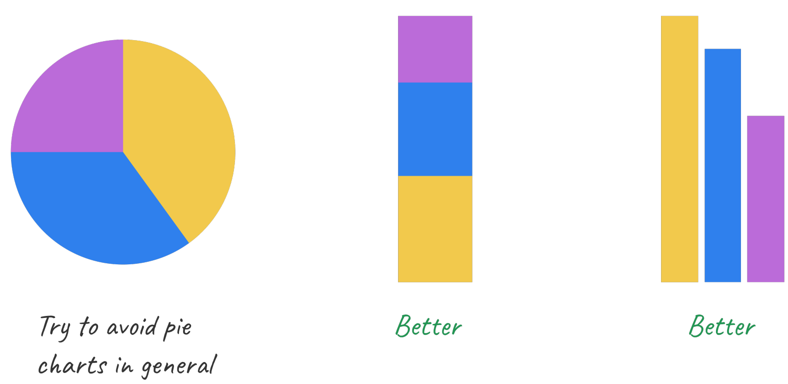


Figure 5.5: Figure 5-4

When your column chart has long x-axis labels that have to be rotated (often 90 degrees) to fit, consider turning the chart 90 degrees so that it becomes a horizontal bar chart. Take a look at Figure 5-5 to see how much easier it is to read horizontally-oriented labels.

If your bar chart shows different categories, consider ordering them, like is shown

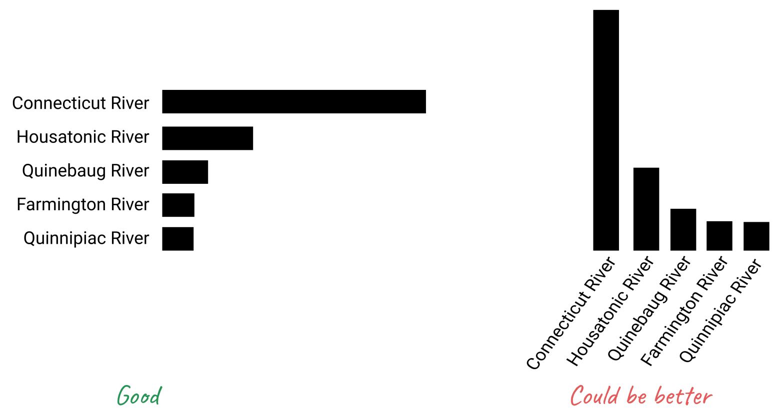


Figure 5.6: Figure 5-5

in Figure 5-6. You might want to sort them alphabetically, which can be useful if you want the reader to be able to quickly ook up an item, such as their town. Ordering categories by value is another common technique that makes comparisons possible. If your columns represent a value of something at a particular time, they have to be ordered sequentially, of course.

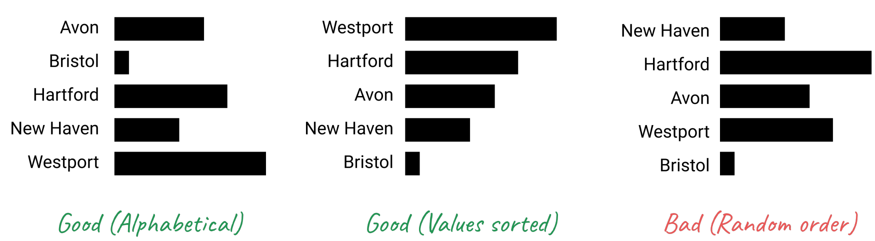


Figure 5.7: Figure 5-6

When labelling a y-axis, choose natural increments that space equally, such as [0, 20, 40, 60, 80, 100], or [1, 10, 100, 1000] for a logarithmic scale. Do not overload your scales. Keep your typography simple, use (but do not overuse) **bolding** to highlight major insights.

The use of color is a complex topic, and there are plenty of books and research devoted to it. But some principles are fairly universal. First, do not use colors just for the sake of it, most charts are fine being monochromatic. Second, remember that color has meaning. When people see the color *red*, they have certain associations, and these associations vary among cultures. In the world of business, red is conventionally used to represent loss, and it would be unwise to use this color to show profit. Make sure you avoid random colors.

Whatever colors you end up choosing, they need to be distinguishable (otherwise what is the point?). Do not use colors that are too similar in hue. Certain color combinations are hard to interpret for color-blind people, like green/red or yellow/blue, so be very careful with those. Figure 5-7 shows some good and bad examples of color use.

Figure 5.8: Figure 5-7

If you follow the advice, you should end up with a de-cluttered chart as shown in Figure 5-8. Notice how your eyes are drawn to the bars and their corresponding values, not bright colors or secondary components like the axes lines.

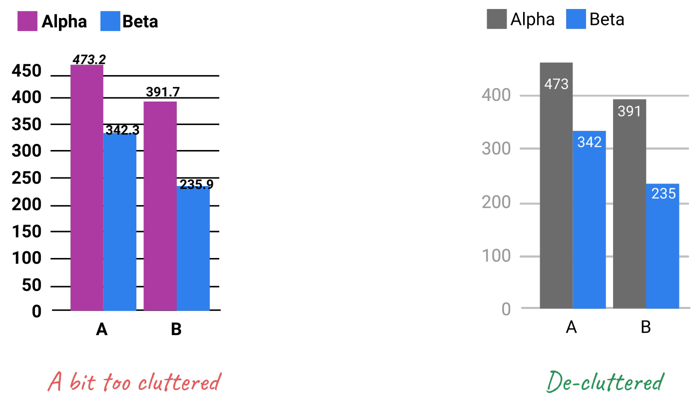


Figure 5.9: Figure 5-8

Google Sheets Charts

Google Sheets (https://sheets.google.com) is a well-known spreadsheet program that allows creating basic charts using intuitive drag-and-drop interface. Most people who create charts with Google Sheets export them as static *png* images. But in fact these interactive charts can be easily embedded in your website.

In this section, we will look at creating column and bar charts that are separated, grouped, and stacked. We will also look at makig pie, line, area, and scatter charts, and learn to visualize three-dimentional data using bubble charts.

As most easy-to-use tools, Google Sheets has its shortcomings when it comes to charting. You might find yourself with too little control over the appearance option. Your won't have much control over your scatterplot tooltips. You won't be able to cite or link to source data inside the chart, but it won't be possible. You won't be able to annotate to highlight items inside charts. But you *will* be able to create good-looking interactive visualizations inside your spreadsheet *quickly*.

Tip: For an overview of charts and graphs in Google Sheets, visit this help page.

Column and Bar Charts with Google Sheets

Column and bar charts are some of the most common types of charts in data visualization (column charts are just vertical bar charts). They are used to compare values across categories.

In this tutorial, we will use three small datasets to build interactive separated, grouped, and stacked bar charts in Google Sheets:

- Obesity in the US (by US CDC, and StateOfObesity.org project)
- High-Caolorie Fast-Food Items
- Global Database on Body Mass Index by World Health Organization

You will need a Google account (it's free).

If this is an e-book, you should be able to interact with the charts in this tutorial. Hover over data points to see tooltips with additional data.

Grouped Column and Bar Charts

Figure 5-9 shows differences in obesity between men and women in three age bracket. If you read this book electronically, you should be able to hover over columns and see tooltips with data.

TODO: FIGURE 5-9 View data from CDC and StateOfObesity.org

The following steps will help you recreate this interactive chart.

1) Right-click to open link in new tab: Google Sheet Column chart with grouped data template
2) Sign in to your Google Drive or sign up for a free account
3) Select File > Make a Copy to save your own version to your Google Drive. Screenshot from Figure 5-10 shows the relevant item in the File dropdown menu.

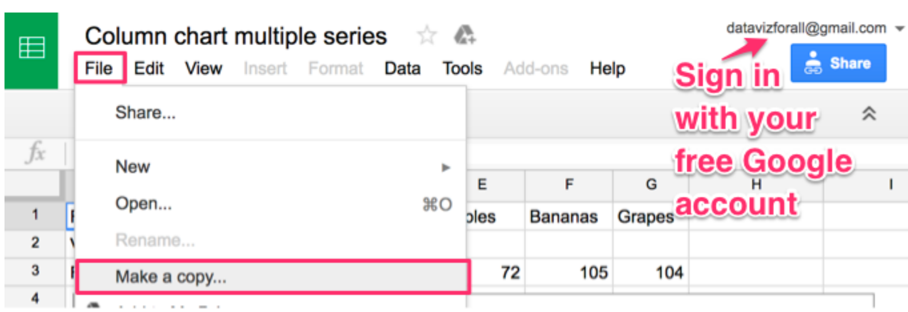


Figure 5.10: Figure 5-10: Sign in to Google and File > Make a Copy

4) To remove the current chart from your copy of the spreadsheet, select it and press the *delete*.
5) Format your data as shown in Figure 5-11. Each row is a data series, which displays as a separate color in the chart.

	A	B	C
1	Age Range	Men	Women
2	20 - 39	31.6	37
3	40 - 59	37.2	44.6
4	60 and over	37.5	39.4

Figure 5.11: Figure 5-11: Grouped column chart data table

6) Use your cursor to select only the data you wish to chart, then select *Insert > Chart*, like in Figure 5-12.

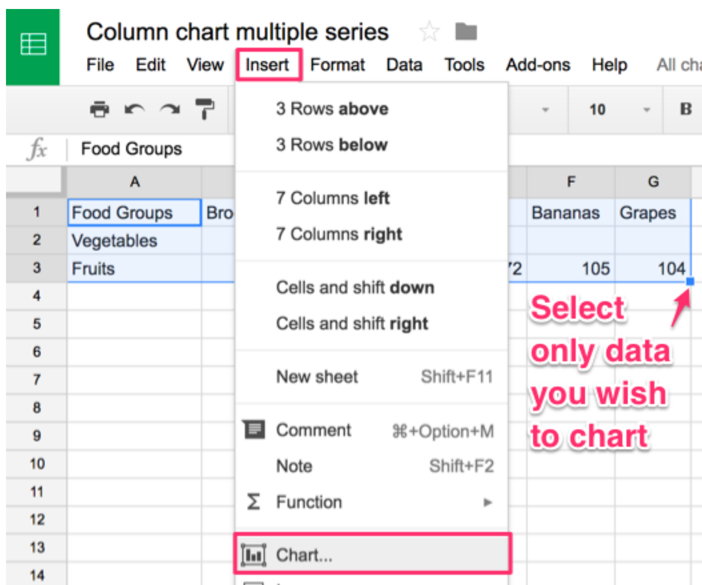


Figure 5.12: Figure 5-12: Select data and Insert > Chart

7) In the Chart Editor > Recommendations tab, choose your preferred Column chart (or horizontal Bar chart if you have longer labels), or see more options in Chart Types tab as per Figure 5-13. Press the Insert button when done.

8) To customize title, labels, and more, choose *Edit chart* from the menu in the upper-right corner of the chart. as shown in Figure 5-15.

9) To make your data public, select Share button in the upper-right corner > Advanced, then Change from Private to Public On the Web, with Anyone Can View.

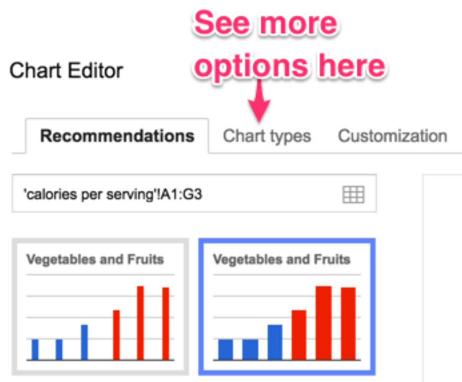


Figure 5.13: Figure 5-14: See more options in Chart Types tab

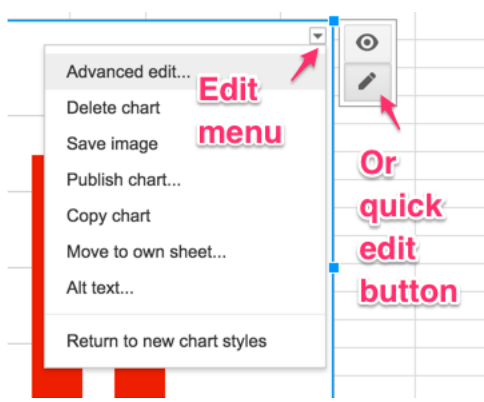


Figure 5.14: Figure 5-15: Customize with editing controls

10) To embed your chart in another website, click the upper-right chart editing controls, select *Publish chart*, select Embed, and press the Publish button. See Chapter 7 of this book to learn what to do with the generated iframe code.

Note: Currently, there is no easy way to cite or link to your source data inside a Google Sheets chart. Instead, cite and link to your source data in the text of the web page. Remember that citing your sources adds credibility to your work.

Separated Column and Bar Charts

When you visualize individual, independent categories, you wouldn't want to group charts. Instead, you want bars (columns) to be separated.

Figure 5-16 shows calorie counts of fast food items for two restaurant chains, Starbucks and McDonald's. Unlike Figure 5-9, here the bars are spaced away from each other.

TODO: FIGURE 5-16

View data from Starbucks and McDonalds

The only difference between making a grouped vs separated bar chart is how you structure your data. To make Google Sheets separate columns, you will need to leave some cells blank, like in Figure 5-17. Other than that, the steps remain the same.

	A	B	C
1	Fast Food items	Starbucks	McDonalds
2	Mocha Frappuccino (24-ounce, 2% milk, whip cream)	500	
3	White Hot Chocolate (20-ounce, 2% milk, whip cream)	710	
4	Big Mac		540
5	Double Quarter Pounder with cheese		770

Figure 5.15: Figure 5-17: Bar chart data table

If you want to get started with the fast-food example, right-click to open this link in a new tab: Google Sheet Column chart with separated data template.

Stacked Column and Bar Charts

Stacked column and bar charts can be used to compare sub-categories. They can also be used to represents parts of a whole instead of pie charts.

The stacked column chart in Figure 5-18 compares the percentage of overweight residents across nations.

TODO: FIGURE 5-18 View data from WHO and CDC

To create a stacked bar chart, you need to choose Chart Type > Stacked column chart (or Stacked bar chart) in the Chart editor window. Structure your data as shown in Figure 5-19. Each column is a new series with its own color. To get started with the Body Mass Index example, begin by opening this link in a new tab: Google Sheets Stacked column chart template.

	A	B	C	D
1	Nation	Underweight	Normal weight	Overweight
2	United States	2	35.2	62.8
3	South Africa	8.6	46.2	45.1
4	Italy	3.4	52.6	44
5	Iran	5.7	51.5	42.8
6	Brazil	4	55.4	40.6
7	South Korea	4.7	63.2	32.1
8	India	32.9	62.5	4.5

Figure 5.16: Figure 5-19: Stacked column chart data table

Histograms

Histogram is a type of bar chart that represents distribution of items, whether numerical or categorical. To bulid a histogram, you need to assign each data point to one of the non-overlapping *buckets* (or *bins*).

Let's say you want to know what time of day are you more likely to get an email. One approach would be to download metadata about all emails you received in 2020, and assign them to a bucket between 0 and 23 according to the email hour. Hours will become your bins, and email counts will be your frequency data. Then your final dataset can look something like:

```
Hour  Emails
0       12
```

```
1        11
2        7
...
13     82
14     103
15     105
16     74
17     53
...
23     22
```

You can now make a histogram. The good news is, Google Sheets considers histograms to be regular column charts, so you should be able to use a previous tutorial to make one.

Hint: Select two columns with the data you want to visualize, and go to *Insert > Chart*. In the Chart editor window, in *Setup* tab, select *Chart type > Column chart*.

Figure 5-20 shows the resulting histogram.

TODO: FIGURE 5-20

If you want to reuse our fictional dataset from the example, make a copy of this spreadsheet.

If you want to have a less detailed histogram, you can combine hours into greater bins, for example *Morning*, *Afternoon*, *Evening*, and *Night* to cover the hours of 6–11, 12–17, 18–23, and 0–5, respectively. Then your dataset will look like:

```
Time of Day,Emails
Morning,353
Afternoon,497
Evening,279
Night,37
```

Bins in a histogram normally span the entire range of values of your dataset. This way no data is being left out. We recommend you use bins of the same size (liek 24 1-hour bins, or four 6-hour bins) to ensure the reader can compare across bars.

Pie, Line, and Area Charts with Google Sheets

Pie Chart

Best to show parts of a whole, but hard to estimate size of slices.

Try it – to come

Tutorial - to come

Line Chart

Best to show change over time with continuous data.

Try it: In this line chart, the level of chicken (shown in orange) rises steadily and surpasses beef (red) and pork (blue). Float your cursor over lines to view data details.

View source data from USDA

Tutorial:

- Begin by opening this link in a new tab: Google Sheet Line chart template
- Follow most of the same steps in first tutorial above.
- Format your data in a similar way as shown below. Each column is a data series, which displays as a separate color in the chart.

	A	B	C	D
1	Year	Beef	Pork	Chicken
2	1910	48.5	38.2	11
3	1920	40.7	39	9.7
4	1930	33.7	41.1	11.1
5	1940	37.8	45.1	10
6	1950	44.6	43	14.3

- In the Chart Editor > Recommendations tab, choose Line chart, or see more options in Chart Types tab.

Stacked Area Chart

Best to show part-to-whole relationships that change over time.

Try it: to come

Tutorial: to come

Scatter and Bubble Charts with Google Sheets

Follow these tutorials to create different types scatter and bubble charts with Google Sheets

Scatter chart

Best to show relationships between two series of data. Also called an XY chart, because each point represents a coordinate value plotted along the horizontal x-axis and the vertical y-axis.

Try it: This scatter chart reveals a downward slope: nations with lower fertility also tend to have higher life expectancy. But remember that a data correlation does not necessarily show causation. Float your cursor over points to view data details. However, the Google Sheet scatter chart only displays static labels for each country, rather than interactive tooltips. See alternative tools below.

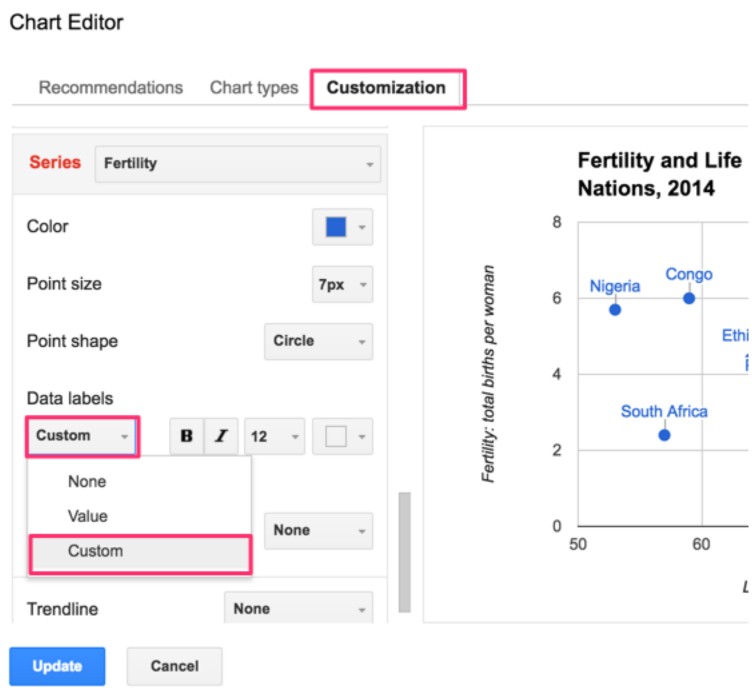
View source data from World Bank

Tutorial:

- Begin by opening this link in a new tab: Google Sheets Scatter chart with static data labels
- Follow most of the same steps in first tutorial above.
- Format your data in a similar way as shown below. The first column (life expectancy) is the x-axis data series, and the second column (fertility) is the y-axis data series. The third column consists of data labels (names of countries).

	A	B	C
1	Life Expectancy	Fertility	Country
2	76	1.6	China
3	68	2.4	India
4	79	1.9	United States
5	69	2.5	Indonesia
6	74	1.8	Brazil

- In the Chart Editor > Recommendations tab, choose Scatter chart, or see more options in Chart Types tab.
- To display static labels for each point, click the upper-right charter corner for Advanced Editing tools > Customization tab, the scroll down to Series > Data labels > Custom, and press Update.



- Since the Google Sheets scatter chart is not ideal, consider using the 3-column bubble chart below, or the Scatter Chart with Tableau Public tutorial in this book.

Bubble chart with 3 columns

Best to show the relationship between two series of data, similar to the scatter chart above, with labels in tooltips.

Try it: This bubble chart shows the same data as above on fertility and life expectancy. Float your cursor over each bubble to reveal a tooltip with the country name and the two data points.

View source data from World Bank

Tutorial: - Begin by opening this link a new tab: Google Sheets Bubble chart with 3 columns template - Format your data in a similar way as shown below, with three columns in this order: - A: label for each bubble - B: numeric data on horizontal x-axis - C: numeric data on vertical y-axis

	A	B	C
1	Country	Life Expectancy	Fertility
2	China	76	1.6
3	India	68	2.4
4	United States	79	1.9
5	Indonesia	69	2.5

- Follow most of the

same steps in the first tutorial above. - In the Chart Editor, skip the Recommendation tab, select the Chart Types tab, then choose Bubble

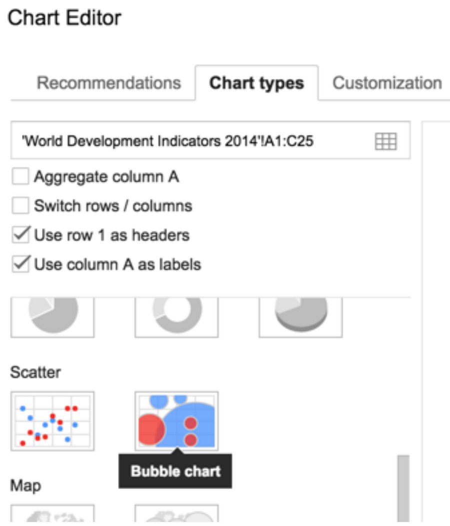
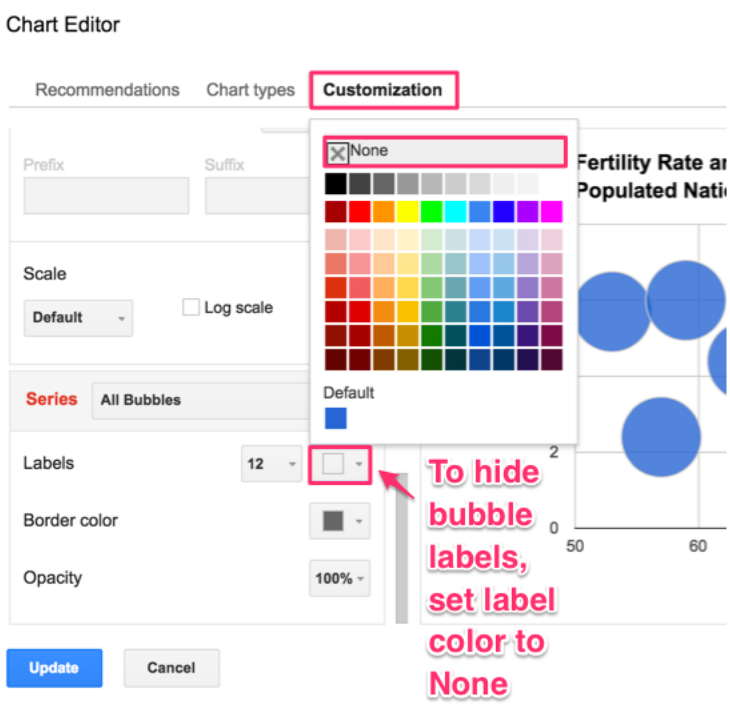


chart (near Scatter chart). - Labels will appear on each bubble by default. To hide labels initially, so that they appear only in the interactive tooltips when floating the cursor over data, customize your chart. Click the editing controls in the upper-right corner, scroll down to Series, and change Labels > Color > None.



- Unfortunately, there is no easy way to reduce all bubbles to a uniformly smaller size. See the Google Sheets Bubble chart with 5 columns below, or create a Scatter Chart with Tableau Public in this book.

Bubble chart with 5 columns

Best to show the relationship between three or four series of data. Similar to a scatter chart, but with bubble size and color to represent additional variables.

Try it: This bubble chart shows fertility and life expectancy for a subset of the nations above, with population (shown by bubble size) and region (shown by bubble color). Float your cursor over bubbles to view data details.

View data from World Bank

** Tutorial ** - Begin by opening this link a new tab: Google Sheets Bubble chart with 5 columns template - Follow most of the same steps in the tutorials above. - Format your data in a similar way as shown below, with 5 columns in this order: - A: label for each bubble - B: numeric data on horizontal x-axis - C: numeric data on vertical y-axis - D: text data to represent bubble color (each category will appear as a new color, or leave blank to display all as one color)

	A	B	C	D	E
1	Country	Life expectancy	Fertility	Region	Population
2	China	75.8	1.56	Asia	1364270000
3	India	68.0	2.43	Asia	1295291543
4	United States	78.9	1.86	North America	318907401
5	Japan	83.6	1.42	Asia	127131800
6	Germany	80.8	1.39	Europe	80982500

- E: numeric data to represent bubble size
- Labels will appear on each bubble by default. To hide labels in the default display (and show them only in the interactive tooltips when floating the cursor over data), see the 3-column bubble chart tutorial above.

Learn more

See additional chart types in this Google Sheets help page

Create Charts with Tableau Public

This book includes tutorials to create interactive charts with Tableau Public https://public.tableau.com. Free download requires email signup.

- Create an XY Scatter Chart with Tableau Public
- Create a Filtered Line Chart with Tableau Public

Tool Review

- Pros
  - Easy-to-learn tool for basic charts, with power to create more advanced visualizations
  - Tableau Public (free version) includes most features found in Tableau Desktop (US $999+)
  - Connect to multiple data formats: Text (CSV), Google Sheets, Excel, and more
  - Combine multiple visualizations and tell stories with dashboard and story point features
- Cons
  - Only available as a downloadable application for Mac or Windows
  - New users may be overwhelmed by extensive options
  - Saving your work online makes it public (hence the name Tableau Public)
  - Limited support for maps below the nation or state levels
  - Dependent upon Tableau web servers

Learn more

- Embed Tableau Public on Your Website chapter in this book
- Tableau Public Resources, with how-to videos and sample data https://public.tableau.com/en-us/s/resources
- Tableau Public Support page https://www.tableau.com/support/public

Create XY Scatter Chart with Tableau Public

An interactive scatter chart shows the relationship between two variables by displaying a series of XY coordinates. Readers can float their cursor over points to view specific details. The chart below, which illustrates the strong relationship between Connecticut school district income and test scores, was created with the free downloadable tool for Mac and Windows, Tableau Public http://public.tableau.com.

Try it

Video with step-by-step tutorial

1) Read the Tableau Public tool review in this book, then download and install the free application on a Mac or Windows computer from http://public.tableau.com. Requires a free account.

2) Click the link and Save to download the sample file to your computer: ct-districts-income-grades-2009-13 in Excel format.

3) Open the sample file to view three columns: district, median household income, and grade levels (above/below national average for 6th grade Math and English test scores). The Notes tab explains how this data is based on the work of Sean Reardon et al. at the Stanford Education Data Archive, Motoko Rich et al. at The New York Times, Andrew Ba Tran at TrendCT, and the American Community Survey 2009-13 via Social Explorer.

Hint: To prepare your own scatter chart data from different sources, see the Match Spreadsheet Columns with VLookup Function chapter in this book.

4) In Tableau Public, click Connect to import the data file from your computer. If you downloaded an Excel file, Connect to Excel. Or if you downloaded a CSV file, Connect to Text. Or click "More…" to connect to Google Sheets.

5) Drag the Data sheet into the Data Source field.

6) In bottom-left corner, below the "Go to Worksheet" reminder, click Sheet 1.

7) Welcome to the Tableau Public Worksheet. Although it may feel overwhelming at first, the key is learning where to drag items from the data tab into the main worksheet. Dimensions are any information that is qualitative or categorical, while measures are quantitative information about the dimensions.

8) Drag the Grade Levels measure into the Rows field.

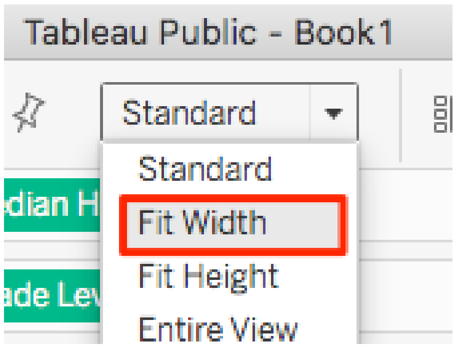
9) Drag the Median Household Income measure into the Columns field. The initial chart will appear as one point, but that's because all of the data is aggregated together. We're not done yet.

10) Drag the District dimension into the lower portion of the Marks area. Now your scatter chart will appear, and float your cursor over each point to view details.

11) Click Sheet 1 to rename the title of your chart.

12) Click the Worksheet menu to Show Caption and type in data sources.

13) Recommended: Click the Standard menu (above Columns) to change view to Fit Width.



14) To publish your chart on the public web, select File > Save to Tableau Public As. Requires signup for a free Tableau account.

15) Give your workbook a meaningful title, since this name will appear in the URL for your published work on the Tableau Public server, and press Save.

16) After publishing your work on the web, Tableau Public will automatically open the web link in your default browser. Click Edit Details to enter more information. Under Toolbar settings, see checkbox to Allow your workbook and its data to be downloaded by others.

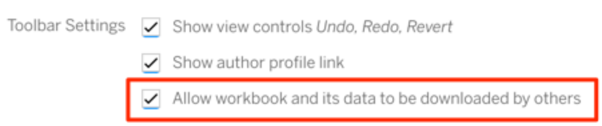


Figure 5.17: Screenshot: Toolbar settings in Tableau Public

Checking this box enables the Download button at the bottom of your published work, which allows users to access your data and workbook, to see how you constructed the visualization.

17) To insert your Tableau Public visualization in your own website, see the Embed On Your Web chapter of this book, and in particular, Embed Tableau Public on your Website.

18) To see all of your published visualizations, go to your Tableau Public online profile, which follows this format:

```
https://public.tableau.com/profile/USERNAME
```

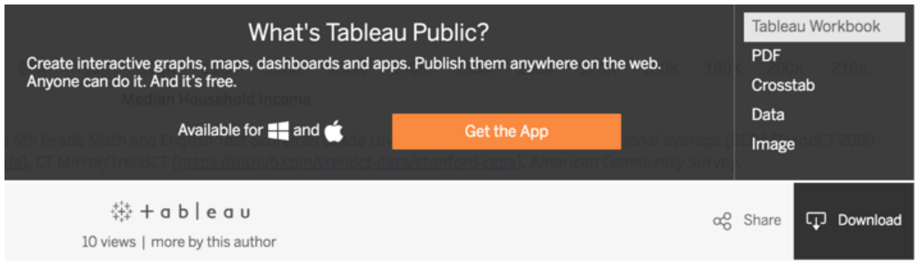


Figure 5.18: Screenshot: Download button in Tableau Public

Learn more

Combine multiple visualizations and tell stories with Tableau Public dashboard and story point features. See Tableau Public Resources, with how-to videos and sample data https://public.tableau.com/en-us/s/resources.

Create Filtered Line Chart with Tableau Public

TODO: Decide whether to keep or not. Originally co-authored with Veronica. An interactive filtered line chart provides checkboxes to turn on/off selected data lines to make specific comparisons, since displaying all of the lines at once would be overwhelming. Readers can float their cursor over each line to identify the school name and data details. We created this tutorial to help a Hartford non-profit education advocacy group compare cohorts of student achievement levels over time across forty schools. You can create your own version with a free downloadable tool for Mac and Windows computers, Tableau Public https://public.tableau.com.

Try it

Or right-click the link to view full-size in a new tab

Video with step-by-step tutorial

1) Read the Tableau Public tool review in this book, then download and install the free application on a Mac or Windows computer from http://public.tableau.com. Requires a free account.

2) Click link and Save file to download to your computer: sample-filtered-line-chart in CSV format. CSV means comma-separated values, a generic spreadsheet format that most data tools can easily open.

Hint: When preparing your own spreadsheet, format your data so that Tableau Public can read it. For example, make sure that Year data is entered as “2007” instead of “1/1/2007”. Leave all blank spaces as-is so that Tableau automatically converts them to “null” values during the data import.

3) In Tableau Public, click Connect to import the data file you downloaded to your computer. If you downloaded a CSV file, Connect to Text. Or if you downloaded an Excel file, Connect to Excel. Or click “More…” to connect to Google Sheets.

4) Your data sheet should automatically appear in Tableau Public. Any blanks will automatically convert to “null.”

5) In bottom-left corner, below the “Go to Worksheet” reminder, click Sheet 1.

6) Welcome to the Tableau Public Worksheet. Although it may feel overwhelming at first, the key is learning where to drag-and-drop items from the data tab into the main worksheet. Dimensions are any information that is qualitative or categorical, while measures are quantitative information about the dimensions. In this example, we are creating a line chart with two dimensions (year and school) and one measure (scores).

7) Drag-and-drop Year into the Column field.

8) Drag-and-drop Schools into the Row field.

9) Drag-and-drop Scores into the middle of the grid.

10) Select Score (but not its drop-down menu), then go to the Analysis menu and turn off Aggregated Measures. We need to do this so that the numbers are displayed individually, and not aggregated by default.

11) In the upper-right corner, go to the Show Me window. (If it is closed, then open it.) Then select Lines (continuous).

12) Initially, each School row appears at its own chart. To blend all of them together into one master chart, drag School to the Marks window and drop it on the Color button. All of the School lines will appear in one chart, with identifying colors.

13) To filter the line chart to display only selected items, go to the Marks window, select the School Cohort drop-down menu, and choose Filter.

14) The Filter window should appear in the far-right side. (If necessary, close the Show Me window to view the Filter window.) Select only a few schools to display by default.

15) Since users can identify schools by turning them on in the Filter window, or floating their cursors to view tooltips for each line, we do not need to show the color legend for each school. In bottom-right School window, select the drop-down menu and choose Hide Card.

16) Confirm or enter text for the axes, title, and caption to describe the data source.

17) To publish your chart on the public web, select File > Save to Tableau Public As. Requires signup for a free Tableau account.

18) Give your workbook a meaningful title, since this name will appear in the URL for your published work on the Tableau Public server, and Save.

19) After publishing your work on the web, Tableau Public will automatically open the web link in your default browser. Click Edit Details to enter more information. Under Toolbar settings, see checkbox to Allow your workbook and its data to be downloaded by others.

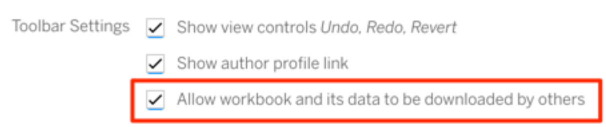


Figure 5.19: Screenshot: Toolbar settings in Tableau Public

Checking this box enables the Download button at the bottom of your published work, which allows users to access your data and workbook, to see how you constructed the visualization.

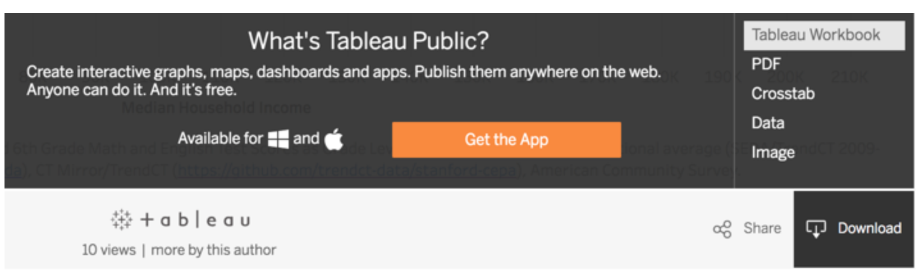


Figure 5.20: Screenshot: Download button in Tableau Public

20) To insert your Tableau Public visualization in your own website, see the Embed On Your Web chapter of this book, and in particular, Embed Tableau Public on your Website.

21) To see all of your published visualizations, go to your Tableau Public online profile, which follows this format:

```
https://public.tableau.com/profile/USERNAME
```

Learn more

Combine multiple visualizations and tell stories with Tableau Public dashboard and story point features. See Tableau Public Resources, with how-to videos and sample data https://public.tableau.com/en-us/s/resources.

See also

- Stephanie D. H. Evergreen, Effective Data Visualization: The Right Chart for the Right Data, (Los Angeles: SAGE Publications, Inc, 2016)
- Stephen Few, Now You See It: Simple Visualization Techniques for Quantitative Analysis, (Oakland, Calif: Analytics Press, 2009)
- Stephen Few, "Save the Pies for Dessert [critique of pie charts]," Visual Business Intelligence Newsletter, 2007, 1–14, http://www.perceptualedge.com/articles/visual_business_intelligence/save_the_pies_for_dessert.pdf
- Stephen Few, Show Me the Numbers: Designing Tables and Graphs to Enlighten, Second edition (Burlingame, CA: Analytics Press, 2012)
- Drew Gourley, How to Use Data Visualization to Win Over Your Audience, (Visage and Hubspot, June 2015), https://visage.co/content/data-viz-win-audience
- Cole Nussbaumer Knaflic, Storytelling with Data: A Data Visualization Guide for Business Professionals, (Hoboken, New Jersey: Wiley, 2015)
- Cole Nussbaumer Knalfic, "An Updated Post on Pies," StoryTelling with Data, February 16, 2017, http://www.storytellingwithdata.com/blog/2017/1/10/an-updated-post-on-pies
- Wayne Lytle, Viz-O-Matic: The Dangers of Glitziness and Other Visualization Faux Pas, 1993 video shared on YouTube, https://www.youtube.com/watch?v=fP-7rhb-qMg
- Isabel Meirelles, Design for Information: An Introduction to the Histories, Theories, and Best Practices Behind Effective Information Visualizations (Rockport Publishers, 2013), http://isabelmeirelles.com/book-design-for-information/

- Tableau, Visual Analysis Best Practices: A Guidebook, n.d., http://www.tableau.com/sites/default/files/media/whitepaper_visual-analysis-guidebook_0.pdf.
- Edward R. Tufte, Beautiful Evidence (Graphics Press, 2006)
- "WTF Visualizations: Visualizations That Make No Sense," 2017, http://viz.wtf.
- xkcd, "University Website," accessed February 12, 2017, https://xkcd.com/773/
- Nathan Yau, "One Dataset, Visualized 25 Ways," FlowingData, January 24, 2017, http://flowingdata.com/2017/01/24/one-dataset-visualized-25-ways/
- Nathan Yau, "Best Data Visualization Projects of 2016," FlowingData, December 29, 2016, http://flowingdata.com/2016/12/29/best-data-visualization-projects-of-2016/

Chapter 6

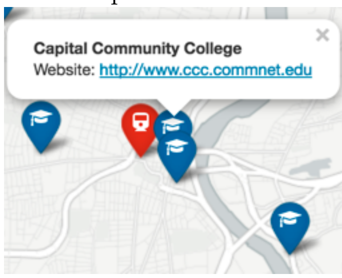
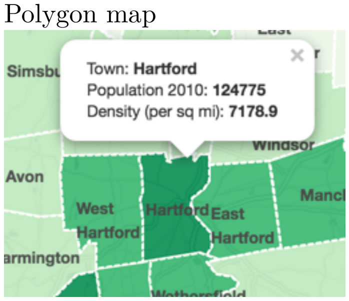
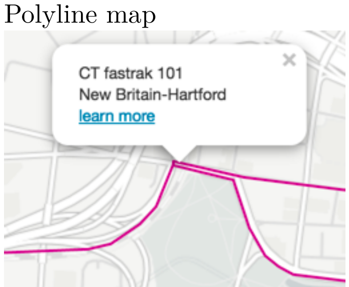
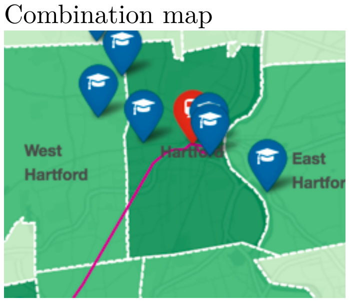
Map Your Data

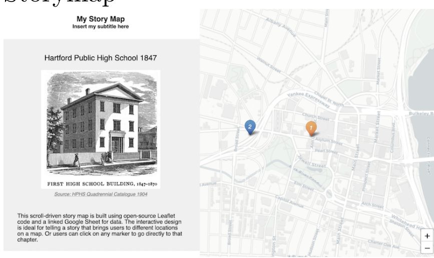
Maps entice readers to explore your data story and develop a stronger sense of place. But good maps require careful thought about how to clearly communicate spatial concepts with your audience. This book features free tools to create interactive maps that you can embed in your website. In this chapter, you will learn how to:

- Practice key principles of map design.
- Choose a map type that matches your data story and format, with tutorial links in the table below. Beginners may start with easy-to-learn tools such as Google My Maps, then move up to more powerful tools, such as Leaflet, which require you to Modify and Host Code Templates with GitHub or other web servers.

See also related chapters in this book:

- Draw and write your data story to capture your ideas on paper
- Improve spreadsheet skills, Find and know your data, and Clean your data
- Transform your map data
- Embed your interactive chart on your website
- Detect bias in data stories, including How to lie with maps
- Tell your data story, including its most meaningful insights and limitations

<table>
<tr><th>Basic map types</th><th>Best use and tutorial chapters</th></tr>
<tr><td>Point map

</td><td>Best to show specific locations, such as addresses with geocoded coordinates, with colors for different categories. Easy tool: Google My Maps tutorialPower tool: Leaflet Maps with Google Sheets and other Leaflet templates</td></tr>
<tr><td>Polygon map

</td><td>Best to show regions (such as nations or neighborhoods), with colors or shading to represent data values. Also known as choropleth map. Easy tool: n/a Power tools: Tableau Public or Leaflet Maps with Google Sheets and other Leaflet templates</td></tr>
<tr><td>Polyline map

</td><td>Best to show routes (such as trails or transit), with colors for different categories. Easy tool: n/a Power tool: Leaflet Maps with Google Sheets and other Leaflet templates</td></tr>
<tr><td>Combination map

</td><td>Best to show any combination of points, polygons, or polylines. Easy tool: n/a Power tool: Leaflet Maps with Google Sheets and other Leaflet templates</td></tr>
</table>

Basic map types	Best use and tutorial chapters
Storymap 	Best for guided point-by-point journey through a historical narrative, with optional photos, audio, or video on an interactive map. Easy tool: Knight Lab's StoryMap, ESRI Story Maps Power tool: Leaflet Storymaps with Google Sheets

TODO:

- heat map
- tab-view map for historical change
- synchronized side-by-side map

Map Design Principles

Ask Before You Map: Before you leap into a mapping project, consider these questions:

Does your data contain geographic information? Common examples:

- Specific locations or addresses (examples: *Trinity College*, or *300 Summit St, Hartford, CT*)
- Latitude and longitude coordinates (example: *41.756, -72.675*)
- Regions that are legally recognized (such as nations, states, counties, census tracts) or that correspond to a boundary map in your possession (such as designated neighborhoods or health districts)

While there are many more types of geographic information, these examples above are the most common. If your data lacks geographic information, or if you do not possess the corresponding boundary information, it may not be possible to map it.

Does location really matter to your data story?

Sometimes a well-designed chart, rather than a map, may be the best way to visualize your data story. Consider these alternatives:

- to show change over time across different locations, consider a line chart

- to show the relationship between two or more datasets across different locations, consider an XY scatter chart or bubble chart

If a map is the best way to tell your data story, then choose an appropriate type. See table of basic map types in this book.

Map Design Principles

1. Understand basic map vocabulary: title, legend, baselayer, marker, popup, tooltip, zoom level, polygon, polyline, source.

2. Add source credits and bylines—with links to view data tables and details—to build credibility and accountability.

3. Choose colors wisely.

    - Use color to logically organize your data. Avoid random colors (Wong pp. 40, 44).
    - Avoid bad combinations from opposite sides of color wheel, such as red/green or yellow/blue (Wong pp. 40, 44).
    - Use contrast (such as color vs gray) to call attention to your data story (Knaflic pp. 87-88)

4. Choose basemaps wisely. Basemaps themselves may contain a lot of information, such as terrain, roads, parks, town names, buildings, etc. They may also use colors that can be distracting to the viewer. Think about the minimum number of elements required in the basemap to tell your story.

Figure 6.1: The view of San Francisco with different basemaps

Design polygon maps with ColorBrewer

One of the most useful tools for creating meaningful polygon (or choropleth) maps is ColorBrewer http://colorbrewer2.org created by Cynthia Brewer, Mark Harrower and the Pennsylvania State University.

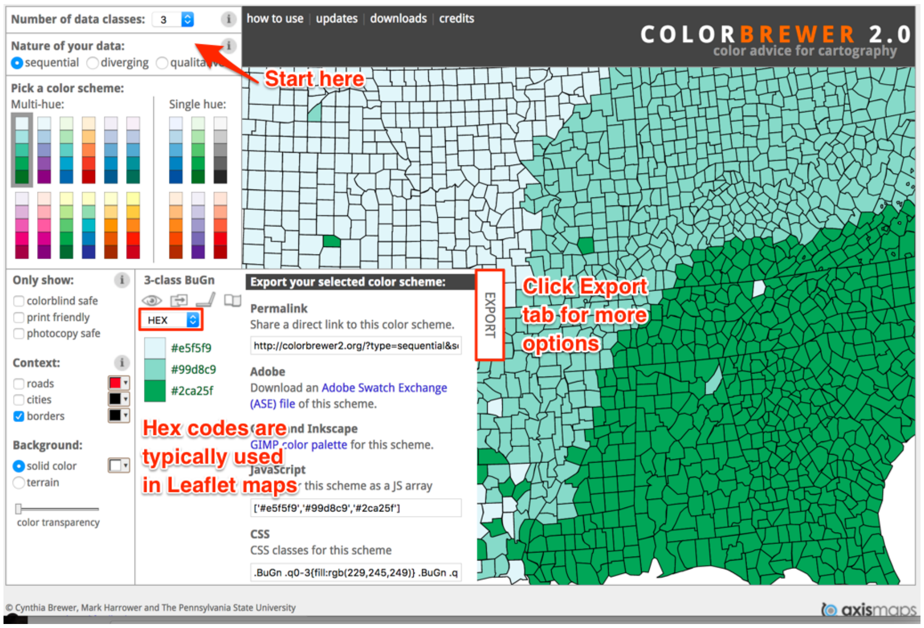


Figure 6.2: Screenshot: ColorBrewer web interface

1) Think about the **number of data classes** (or "dividers" or "buckets"). More does not necessarily mean better. Try different numbers and color schemes, and decide if you (and your audience) can easily distinguish between them.
    - A smaller number sorts your data into fewer buckets, and shows a more **coarse map**, but differences in colored ranges become **more visible**.
    - A larger number sorts your data into more buckets, and shows a more **granular map**, but differences in colored ranges become **less visible**.

2) Think about the **nature of data** you are going to display.

- Sequential: best to show steps from lower values (light color) to higher values (dark color)

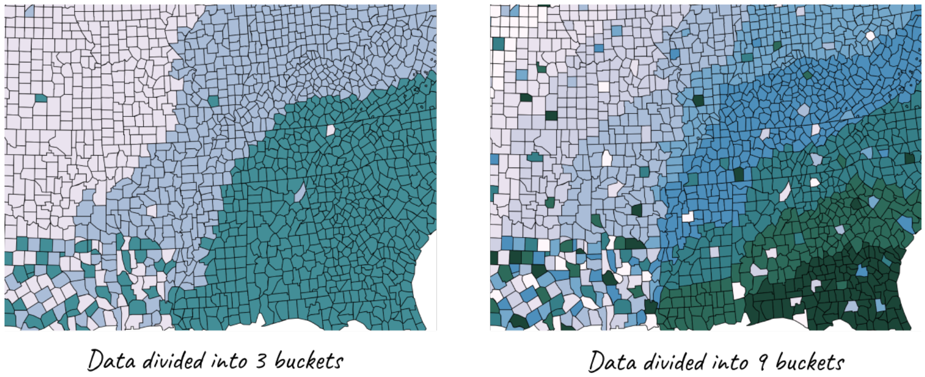


Figure 6.3: Screenshots: ColorBrewer web interface

  - Example: a scale that increases from 1 to 100

- Diverging: best to show extremes (dark colors) around a neutral middle (light color)
  - Example: a scale that highlights extremes from -100 to 0 to 100
- Qualitative: best to show different categories, represented by their own color
  - Example: a map legend of the dominant crop in each area: apples, oranges, bananas

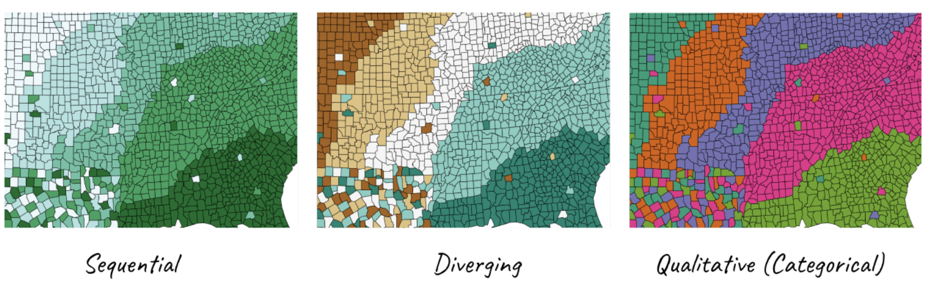


Figure 6.4: Screenshots: ColorBrewer web interface

3) Pick a **color scheme**, with options for colorblind-safe and print-friendly.
   - Think about the ideal format for your audiences. Are readers more likely to view your visualization on a computer screen, or in print, or both?
4) Click the Export tab to view all options. Some Leaflet map templates in this book use specific color names (such as “red” or “darkgreen”) and

some use hexadecimal codes, abbreviated as "hex codes" (such as #ff0000 or #336600). To learn more, use a Color Picker tool, such as https://www.w3schools.com/colors/colors_picker.asp

Beware that polygon map design choices about data classes and colors reflect the biases of the author and the software. Read the Detect Bias in Data Stories chapter in this book, especially How to Lie with Maps

Learn more

- Axis Maps, "The Basics of Data Classification," 2010, http://axismaps.github.io/thematic-cartography/articles/classification.html
- Lisa Charlotte Rost, "Your Friendly Guide to Colors in Data Visualisation," Lisa Charlotte Rost, April 22, 2016, https://lisacharlotterost.github.io/2016/04/22/Colors-for-DataVis/.
- Josh Stevens, "Bivariate Choropleth Maps: A How-To Guide," February 18, 2015, http://www.joshuastevens.net/cartography/make-a-bivariate-choropleth-map/.

Point Map with Google My Maps

TODO: add text

Try It

Explore the interactive point map below, or view the full-screen version, created with Google My Maps https://www.google.com/maps/d/.

Tool Review

- Pros
  - Easy-to learn free mapping tool to import and style point, polyline, and polygon layers and basemap layers
  - Share and collaborate through the Google Drive platform
  - Geocoding error warning
- Cons
  - Limited options to customize map markers
  - Cannot easily create colored polygon maps from data values
  - Cannot extract geocoded data to migrate to another tool

Video with Step-by-Step Tutorial

Let's build a simple point map with sample data, using Google My Maps https://www.google.com/maps/d/. Requires signing up for a free Google Drive account.

1) Click this link and Save to download to your computer: sample-address-data in CSV format. CSV means comma-separated values, a generic spreadsheet format that most tools can easily open. For help with downloading, see this short video tutorial.

2) Open and sign in to Google My Maps https://www.google.com/maps/d/

3) Click the red + symbol to create a new map, which will be saved automatically to your Google Drive folder.

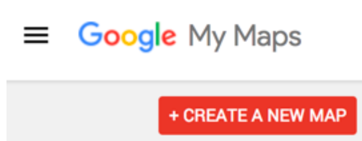


Figure 6.5: Image: Create a new map

4) In the map layers area, click the blue Import link. Drag-and-drop the CSV address data file into the web interface to import it.

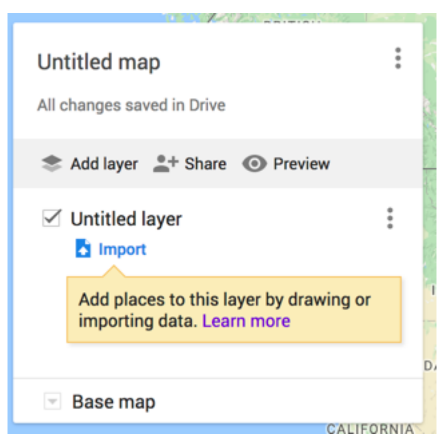


Figure 6.6: Image: Import a data layer

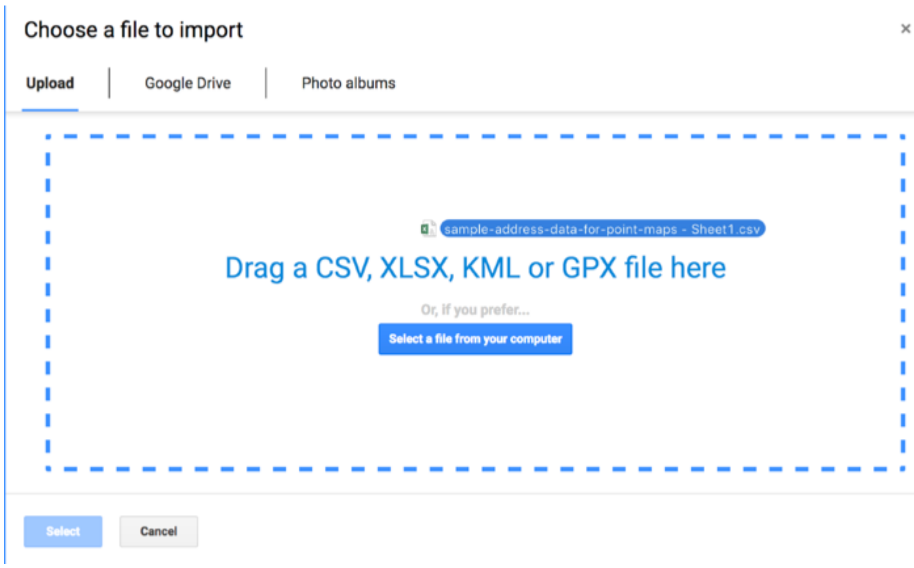


Figure 6.7: Image: Drag-and-drop data into My Maps

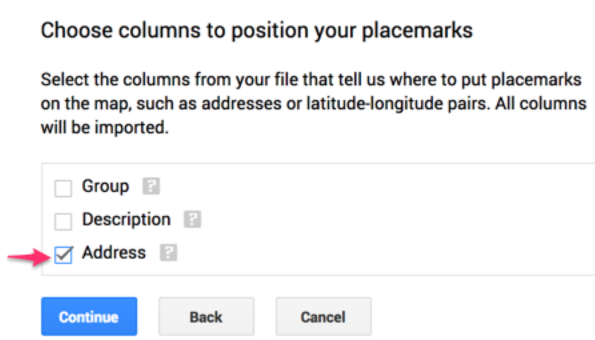


Figure 6.8: Image: Choose columns to position placemarks

5) Choose columns to position your placements. Select "Address" for this sample data, then Continue.

6) Choose a column to title your markers. Select "Description" for this sample data, then Finish.

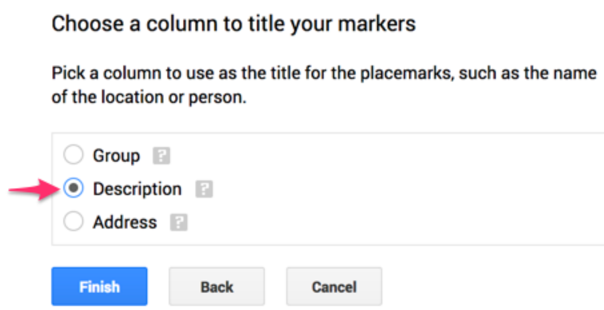


Figure 6.9: Image: Choose column to title markers

7) After My Maps uploads and geocodes your sample data, click Open Data Table to inspect the results.

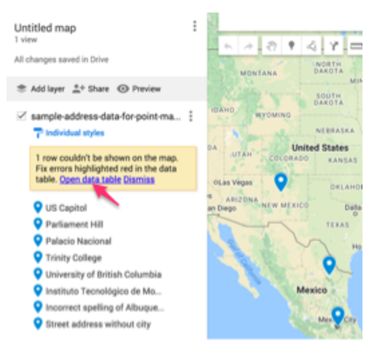


Figure 6.10: Image: Open Data Table to inspect geocoding errors

8) To style the map markers, click Individual Styles. In this sample data, you can select Group Places By > Style By > Group. This will color markers according to the three categories.

9) To publish your map on the web, click Share, add a map title, change from Private to Public on the Web, so that anyone can view your map. Click Save and Done.

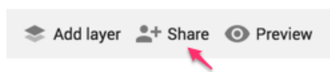


Figure 6.11: Image: Share link

10) To embed the map on your own website, click the three vertical dots next to the map title for more options, and select Embed On My Site. The tool will generate an iframe code for you to copy. For next steps, go to the Embed on Your Web chapters in this book.

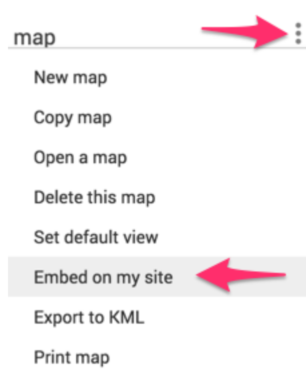


Figure 6.12: Image: Embed map on your site

Learn more

- Google My Maps Help Page https://support.google.com/mymaps/answer/3024396

Point Map with Carto Builder

TODO:

- Test this tool and decide if it still warrants inclusion in this book
- See note about old versus newer Cartobuilder – still relevant?
- if this tool stays in the book, check the iframe below to see if update is needed

Try It

Explore the interactive point map below, or view the full-screen version ,created with Carto Builder https://carto.com.

Tool Review

- Pros:
  - Free and powerful drag-and-drop map tool in the browser
  - Customize point markers and polygon colors by data values
  - Additional features include geographic analysis tools
- Cons:
  - Several steps required to create simple point or polygon map
  - New users may get lost when moving through multiple screens
  - Free account allows only 400 geocodes per month

Video with Step-By-Step Tutorial

Before you begin: This tutorial uses the newer Carto Builder, rather than older Carto Editor tool. Learn more at https://carto.com/learn/guides/intro/migrating-from-carto-editor-to-carto-builder. If you have an old Carto account that has not automatically updated to the new Builder tool, you may need to create a brand-new account to use this tutorial.

Let's build a simple point map with sample data, using Carto Builder https:/carto.com. Requires signing up for a free account.

1) Click this link and Save to download to your computer: sample-address-data in CSV format. CSV means comma-separated-values, a generic spreadsheet format that many tools can easily open.

2) Open Carto in your browser https://carto.com.

3) The Carto Dashboard displays two views: Maps and Datasets. Always begin with Datasets, then move to Maps. (Hint: If your dashboard looks very different than mine, then you might still be using the older Carto Editor, rather than the newer Carto Builder.)

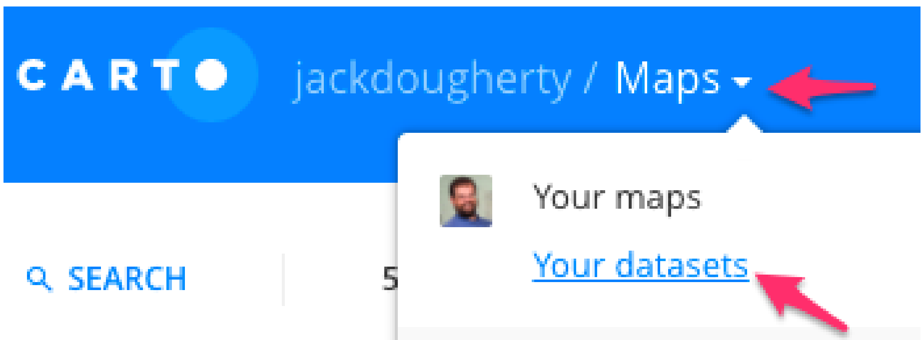


Figure 6.13: Image: Carto Builder dashboard: Begin with Datasets

4) First, connect your dataset, and soon we'll turn it into a map. Click blue button to add New Dataset.

5) Drag-and-drop the CSV sample address data to upload it, and select Connect Dataset. (Be patient. Sometimes this takes more than 30 seconds.)

6) Inspect your connected dataset.

7) Click the blue Create Map button.

8) Click the Edit Your Map button.

9) In your map data layer, click Add Analysis.

10) In the next screen of Analysis options, select Georeference, then click the Add Analysis button.

11) Back in your map data layer, under Georeference options, select Type > Street Addresses (scroll down to the bottom) for this sample data.

12) Under Parameters, for Column Street Address (abbreviated as Col. Street Ad.), select the "address" field for this sample data. Press the Apply button.

13) After Carto has attempted to geocode your address data, click Style This Analysis. Or, go to the map data layer and click the Style tab.

14) In Style options, for Aggregation select none (the default).

15) Under Style options:

- select Fill Number to change circle sizes
- enter a larger size, such as 13, to make our sample points more visible
- select Fill Color to change circle color
- switch from Solid (all points are same color) to By Value, and scroll down to Group (at the bottom) to automatically color by categories for this sample data. (Hint: If you don't see Group in the menu, click somewhere else and try it again.)

16) In the Pop-up tab, select a Window Style, then select boxes in Show Items to display.

17) In the Legend tab, click Select a Style to display information, and your color-coded groups from above should automatically appear on your map. (Hint: A legend may automatically appear after styling your markers by color.)

18) Before publishing your map: If you wish to rename it, do it now by selecting the three vertical dots next to the file name, and select Rename.

19) To publish your map on the web: Next to your map file name, click the blue "back" arrow (NOT your browser back button) to return to the data layer. Click the green Public button, and on the next screen, click the blue Publish button.

20) On the next screen, Get The Link generates a weblink to your map, and Embed It generates an iframe code to insert the live map in your website. For next steps, go to the Embed on Your Web chapters in this book.

21) If you make edits to your map, you must click the blue Update button to republish your map to the web.

Learn more

- Getting Started with Carto Builder https://carto.com/learn/guides/intro/getting-started-with-carto-builder

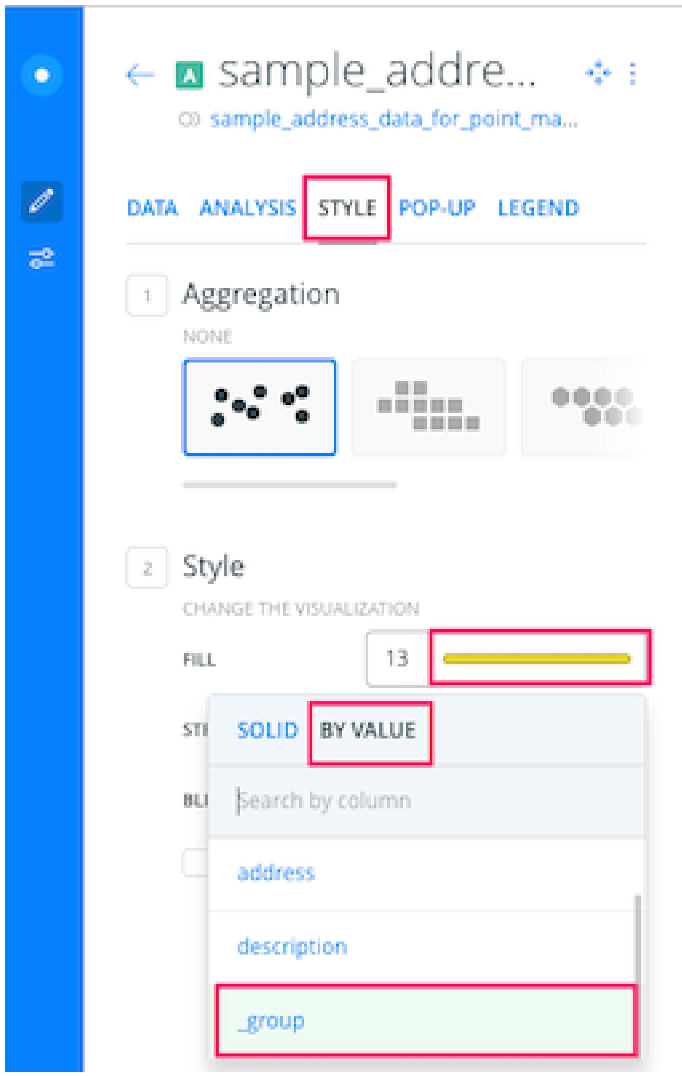


Figure 6.14: Image: Style points by value

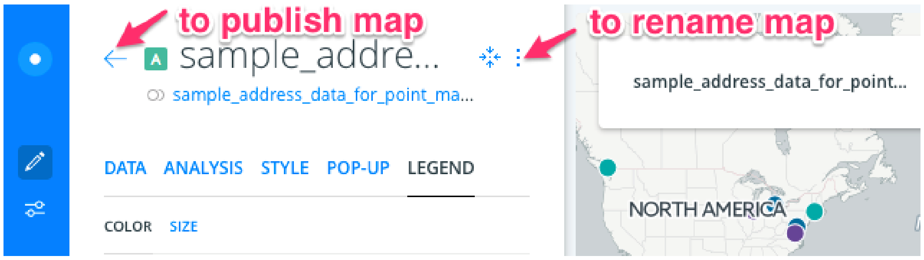


Figure 6.15: Image: Click to rename or publish your map

Filtered Point Map with Socrata Open Data

TODO: decide whether to keep or not; originally co-authored with Veronica.

Open data repositories recently launched by the State of Connecticut and the City of Hartford both use the Socrata platform, which offer user-friendly ways to view, filter, and export data. Also, the Socrata platform includes built-in support to create interactive charts and maps, and to embed them on your own websites. This tutorial demonstrates these features by creating an interactive point map of selected schools from the Connecticut Education Directory in the state data portal. The final product looks like this:

CT Schools Map 2015

Powered by Socrata

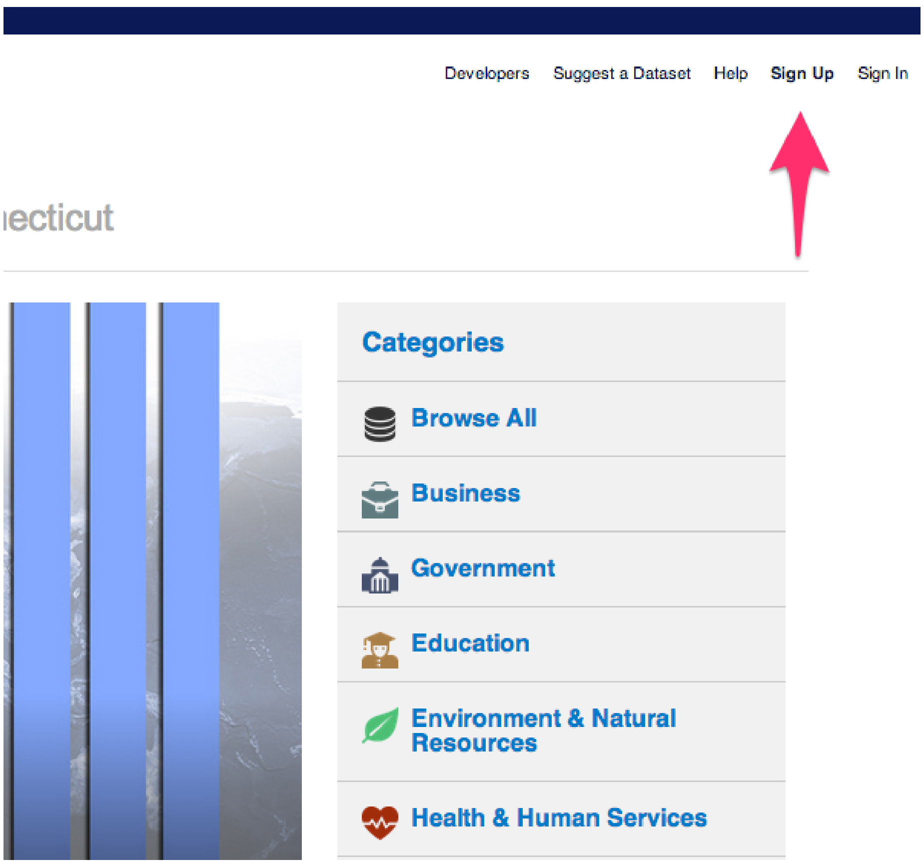
One advantage of creating data visualizations directly on an open data platform is that the chart or map is linked to the data repository. For example, if the Socrata platform administrator updates the data table, then a Socrata dataviz based on that data will be automatically updated, too. This may be especially useful for "live" data that is continuously updated by agency administrators, such as fire, crime, and property data repositories.

But there are limitations to creating your chart or map on an open data repository platform. First, if the agency stops using the platform, or changes the structure of the underlying data, your online chart or map may stop functioning. Second, you are usually limited to using data tables and geographic boundaries that already exist on that platform, since importing your own may not be an option.

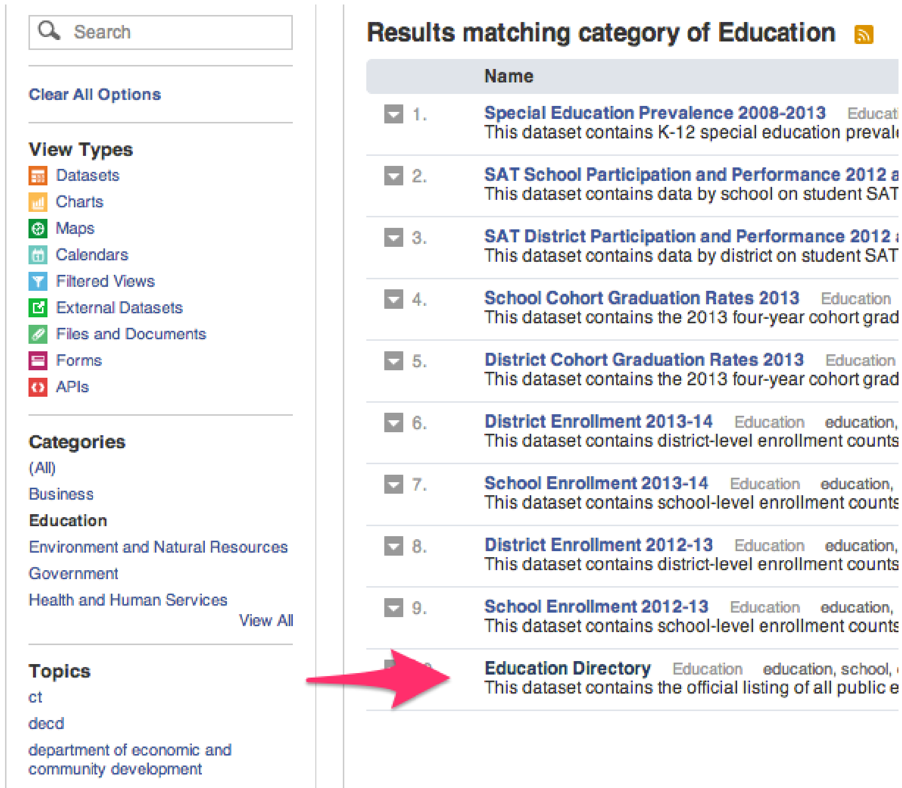
If these limitations concern you, a simple alternative is to export data from the open repository (which means that any "live" data would become "static" data), and import it into your preferred dataviz tool, such as those described in other chapters of this book. A second, more advanced alternative, is to learn how to pull live data from the repository directly into your dataviz, using an Application Programming Interface (API), which requires coding skills that are beyond the scope of this tutorial. To learn more about the Socrata API: https://dev.socrata.com/.

Steps to create a filtered point map

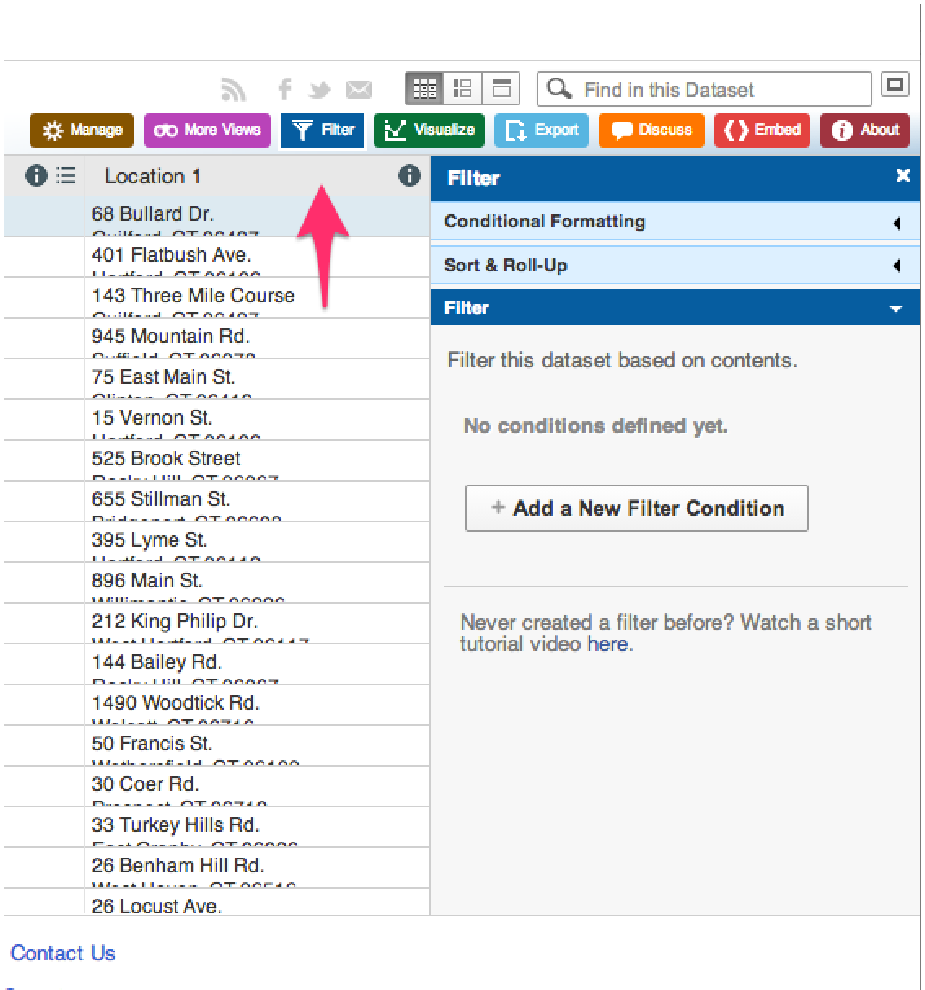
Sign up for a free account ID on any Socrata platform, such as https://data.ct.gov/signup. One account will work on all Socrata sites.



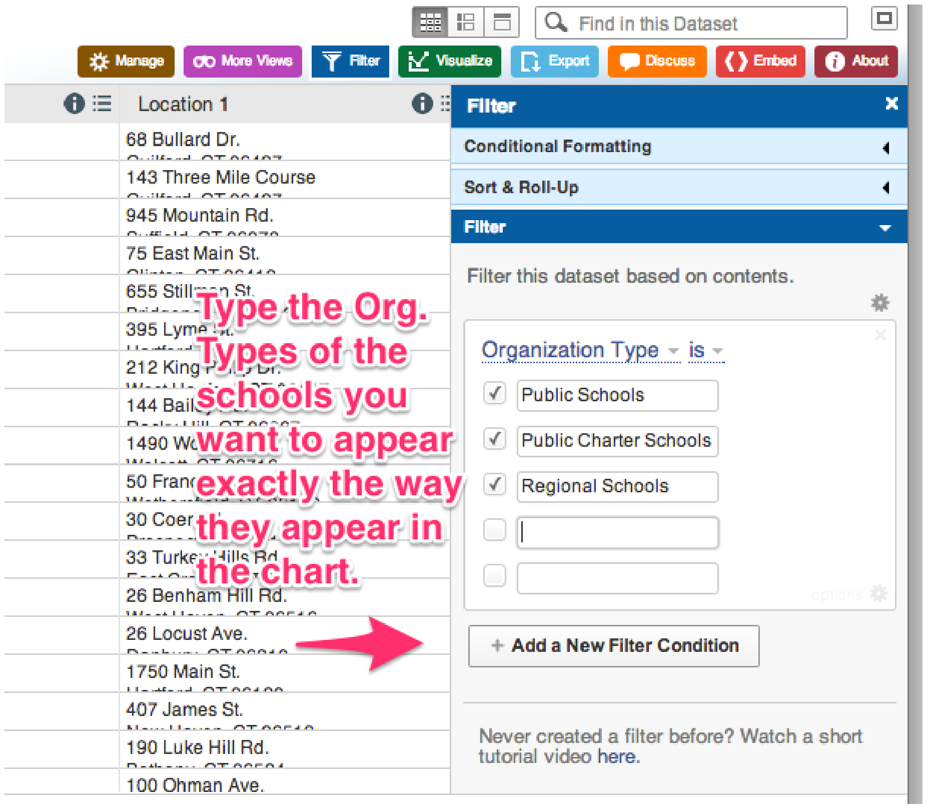
Select your desired dataset in Socrata. In this tutorial, we will use CT Open Data > Education > CT Education Directory. The data table must include a location column that includes geocoordinates. If there is address data but no geocoordinates, then post a suggestion to the Socrata site administrator to add a geocoded column.



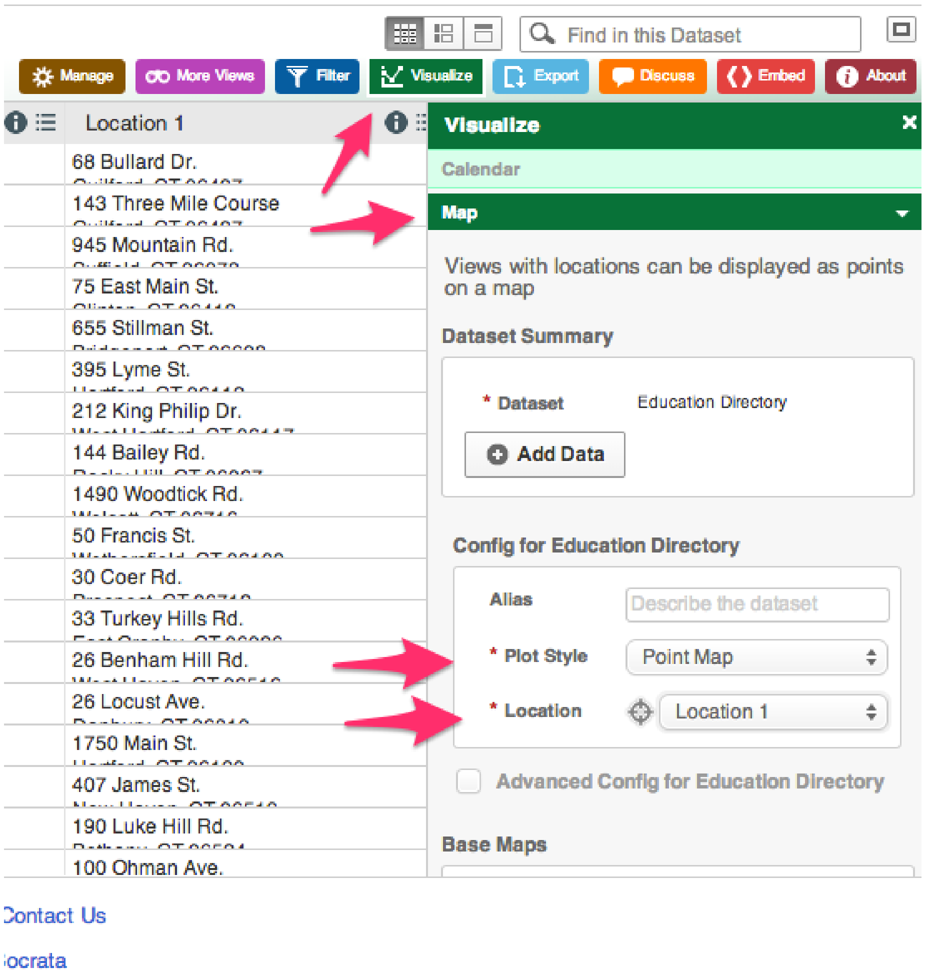
Filter the data to display only the desired rows. The CT Education Directory lists both district offices and school addresses, but for this map we only wish to display the latter. On the top-right corner of the table, click the Filter tab.



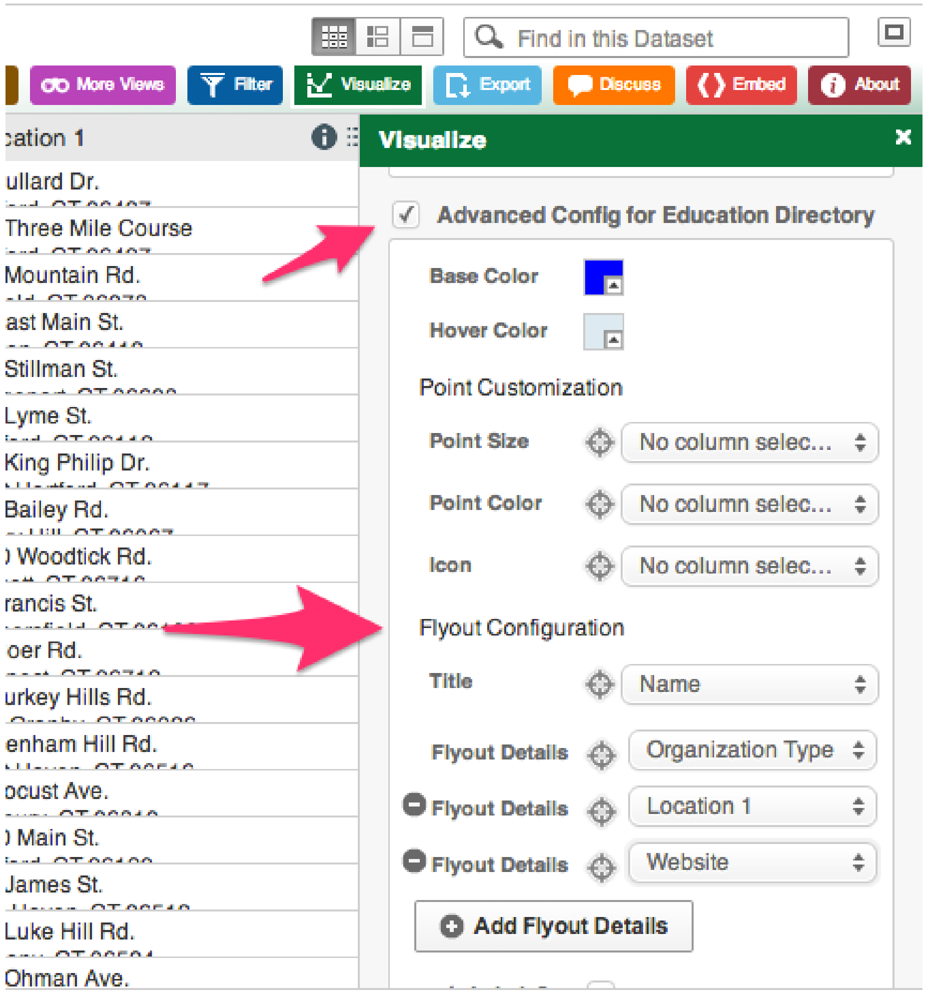
Add a New Filter Condition, which displays only the rows you select. In this tutorial, select “Organization Type” and “is”, then type the exact name from the table, such as “Public Schools.” Be sure to type it correctly or the filter may not work. If you wish to select multiple types, add a new filter condition for each. In this tutorial, we also will filter for other types: Public Charter Schools, CT Technical High Schools, Regional Schools, State Agency Facilities, Endowed and Incorporated Academies Schools, and Regional Education Service Center Schools.



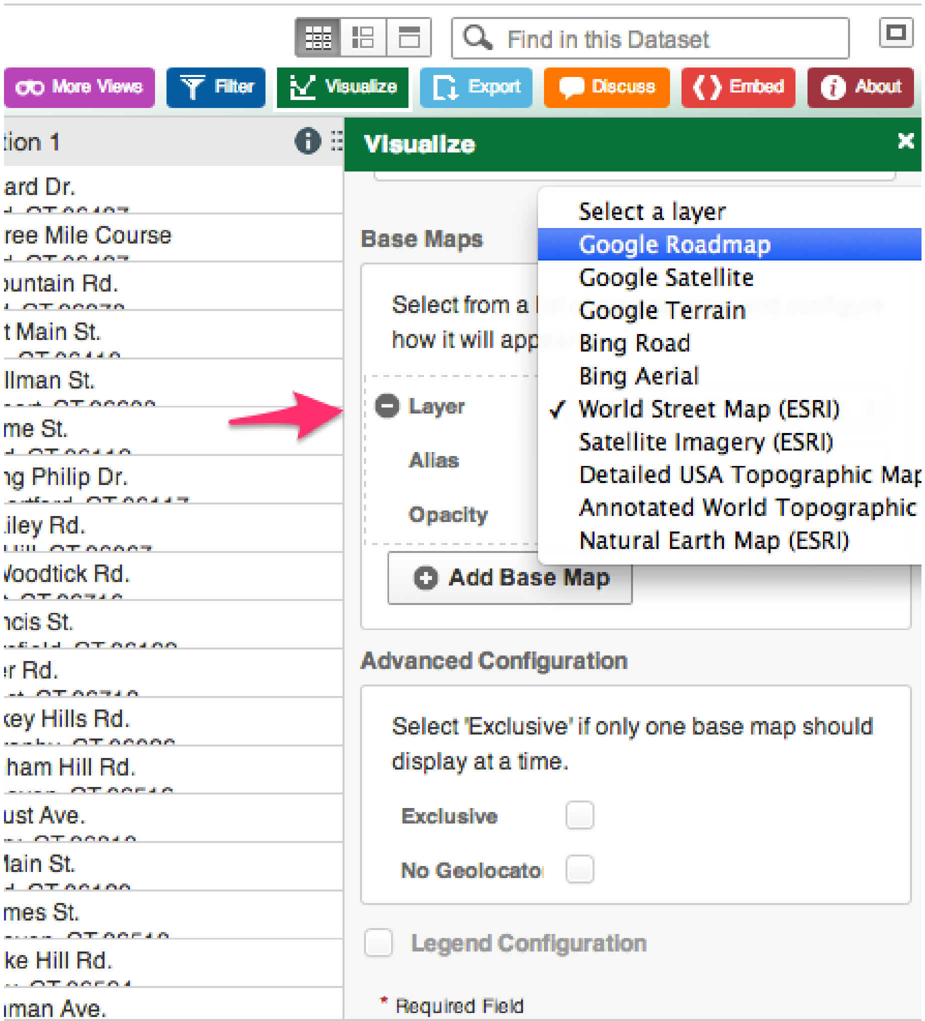
Select the Visualize tab and choose Map, which will display several options. First, under Config for Education Direction, select Point Map as the Plot Style, and choose the Location column to identify the geocoordinates.



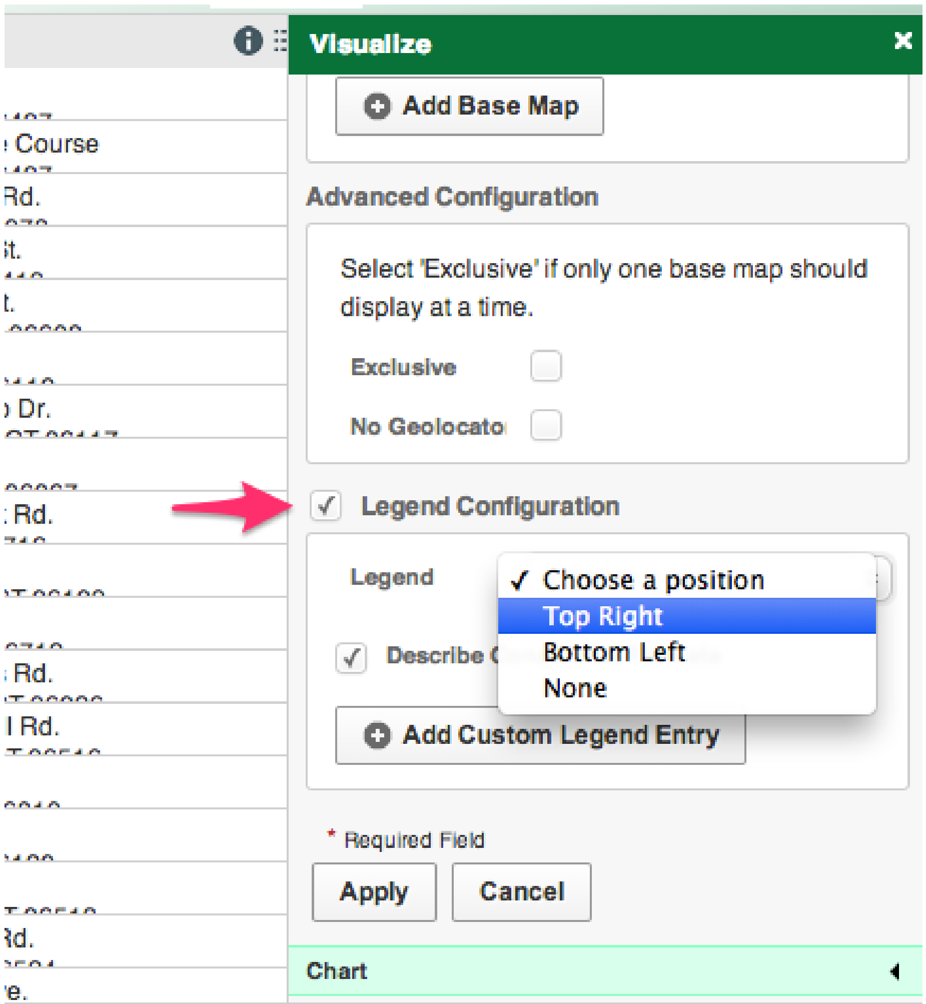
Further below in the Visualize > Map options, select the checkbox for Advanced Config for the Education Directory to edit the Flyout Details (similar to a pop-up information window) that displays details when users click on a map point. Select data items you wish to display, such as Title: Name, and additional Flyout Details: Organization Type, Location I, and Website. Further down, select the "w/o labels" checkbox to avoid displaying the column headers in your flyout details.



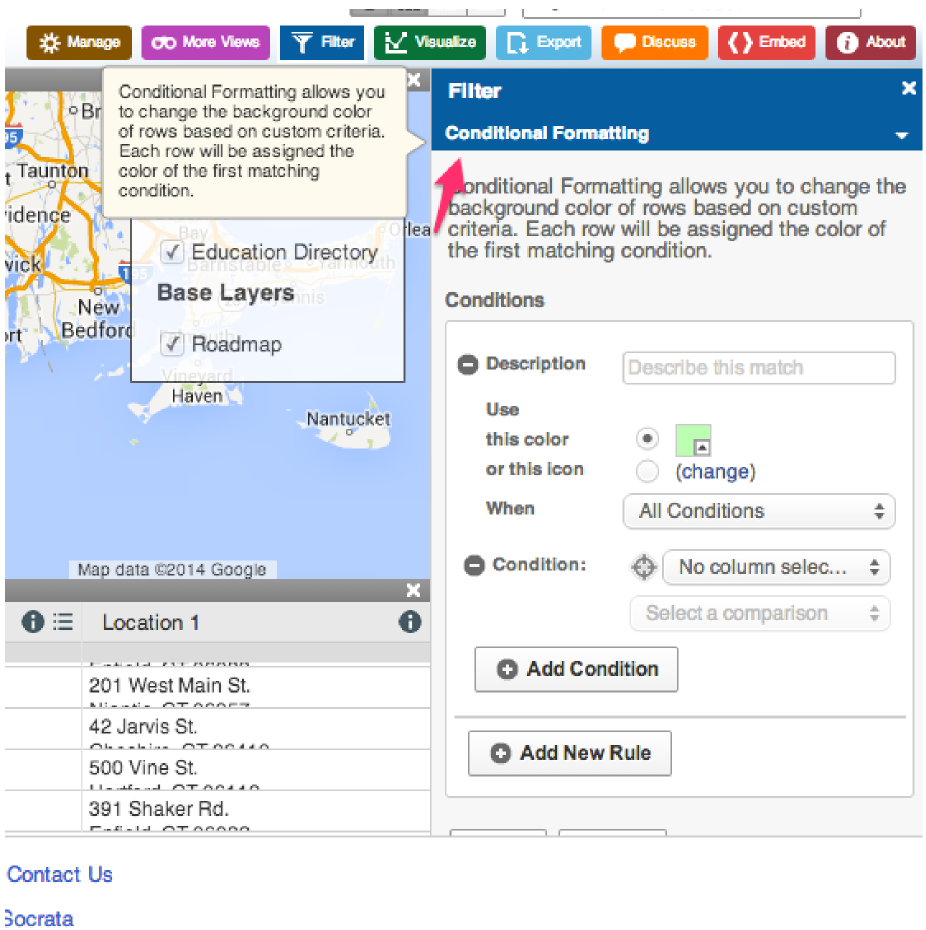
In Visualize > Map > Base Maps, select your desired background map, such as Google Roadmap.



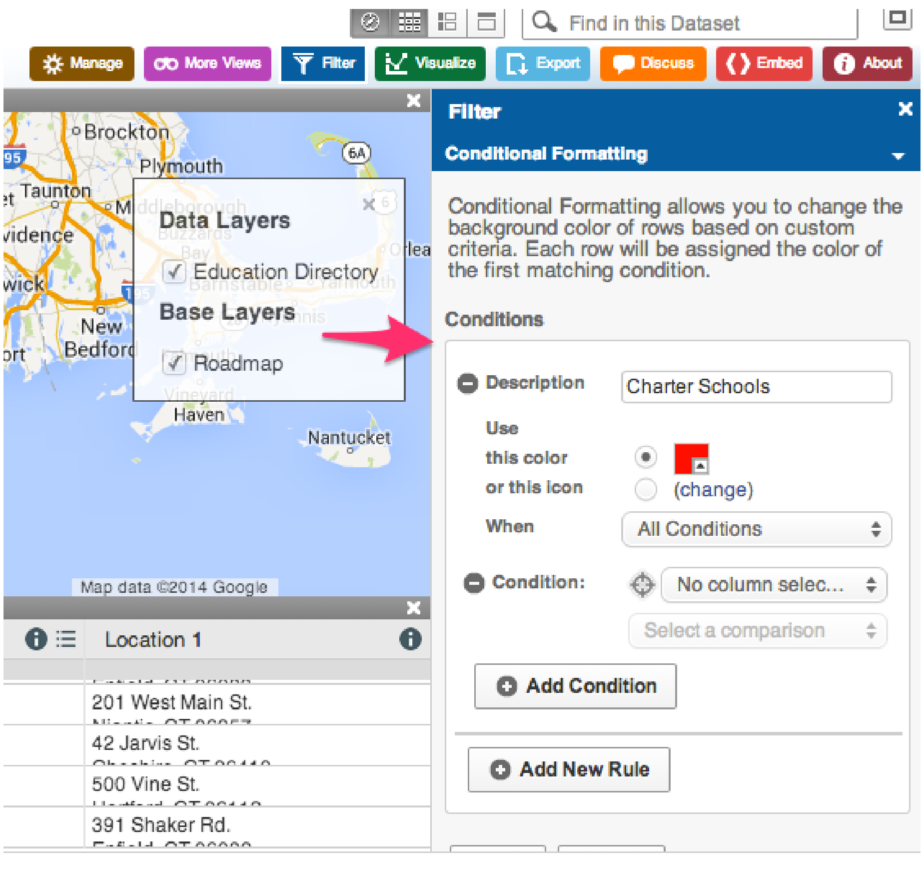
Add a legend to display once you build the map. In the Advanced Configuration area, select the Legend Configuration checkbox and mark its position. After selecting all of these map options, click Apply. Socrata will generate your map with default point colors. Double-check to make sure your data appear, and that your Visualize settings are correct, before moving to the next step.



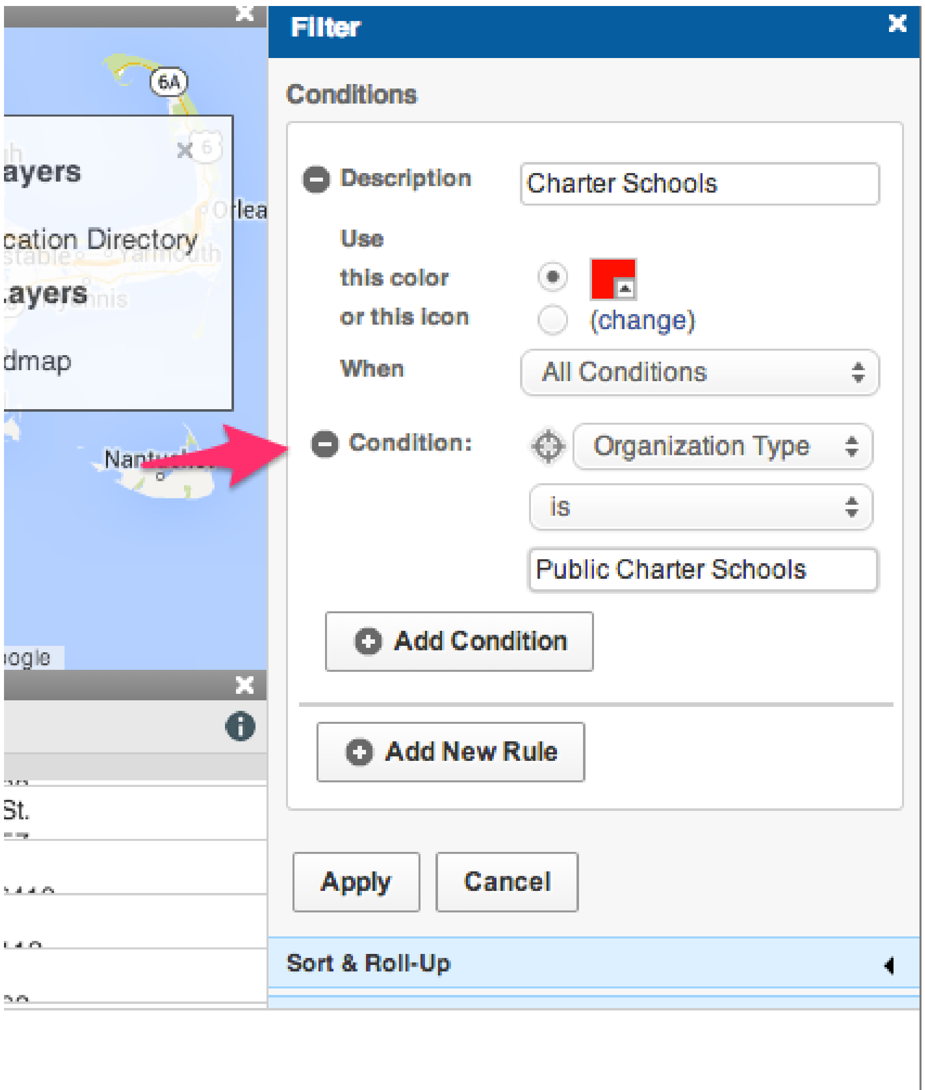
Assign point colors and legend labels by returning to the Filter tab, and select Conditional Formatting. Understand the difference between these two features. Previously, we used Filter to display only selected types of data (in this case, school buildings, rather than district administrative offices). Now, we will use Conditional Formatting to assign color codes and labels to our filtered data.



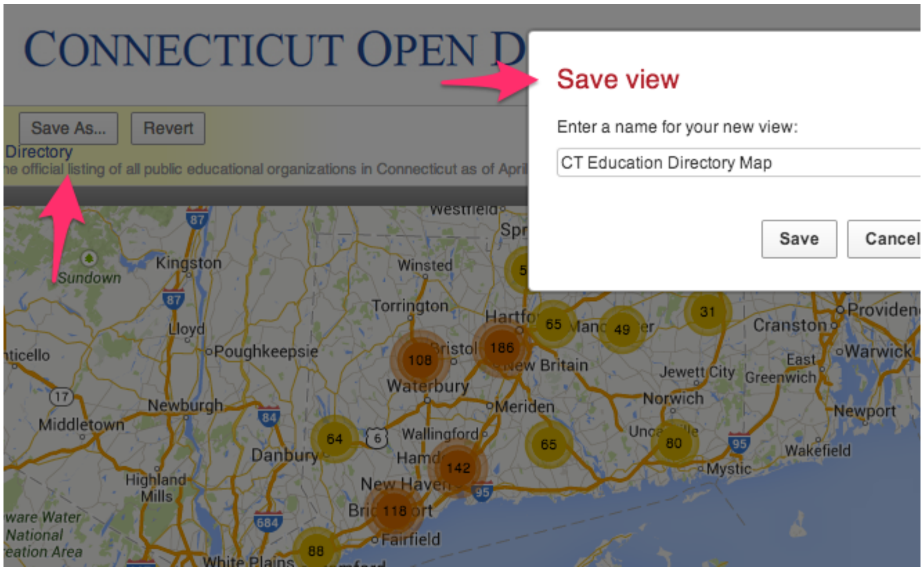
In the Conditional Formatting section, type a name into the Description that you wish to display in the legend. You may type a shorter name than the longer name that appears in the data table, such as “Charter Schools” instead of the longer “Public Charter Schools.” Also, select a color for each Description.



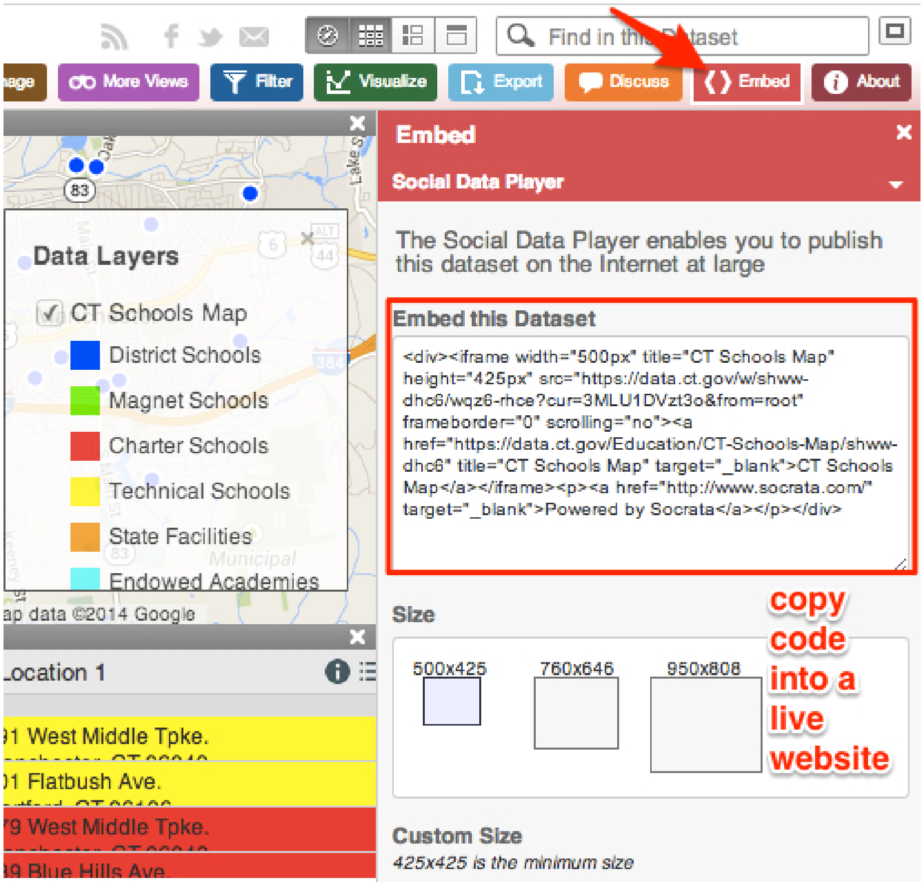
Continue to add Conditional Formatting by defining the data columns. In this example, select "All Conditions Apply," choose "Organization Type" and "Is", then type the category exactly as it appears in the data table (such as Public Charter Schools). For this map of schools in the CT Education Directory data table, we added several more types (Regional Schools, CT Technical High Schools, etc.) and also added a second rule to identify Magnet Schools (where Organization Type is Public Schools, and Interdistrict Schools is 1).



After setting all of your Conditional Formatting, press Apply at the bottom of the tab. Double-check that your visualization appears exactly as you wish, then Save As under an appropriate name. Note that your visualization will become **publicly visible** to other users on the Socrata open data platform, though you have the option to remove it via your individual profile view.



Visualizations created in the Socrata platform produce HTML iframe codes, which allows you to embed the dataviz in your own website. Select the Embed tab to view and copy the code. Then go to the Embed on the Web chapters in this book.



Polygon Maps and Storyboards with Social Explorer

TODO: decide whether to keep or not, since free license terms changed

The Social Explorer free edition http://socialexplorer.com offers one solution to creating colored polygon maps with US Census demographic data. Explore the embedded sample map below.

Advantages

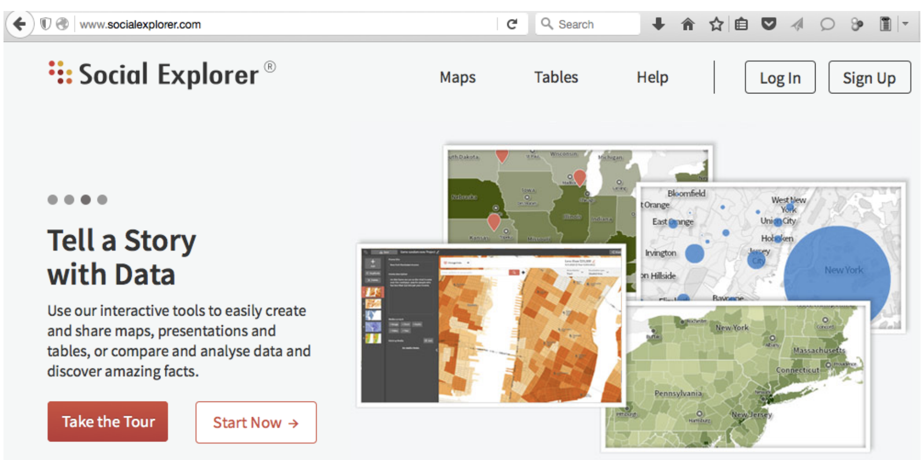
- Quick and easy-to-learn
- Free edition includes basic census data
- Export your static maps into presentation slides
- Share link or embed iframe to your interactive map

Limitations

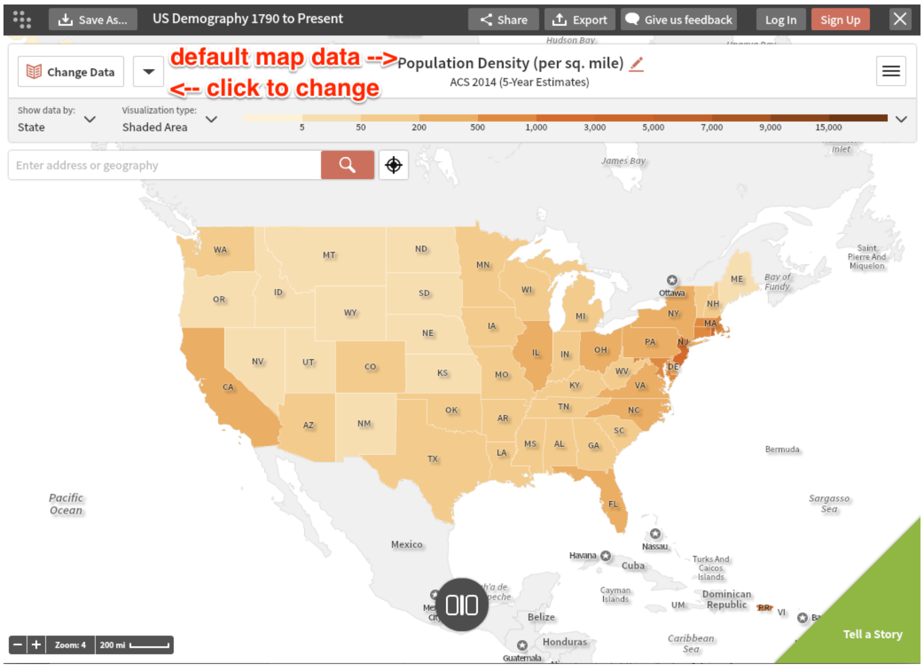
- Maps are limited to the demographic data inside the tool.
- Polygon map boundaries are limited to state, county, census tract. The tool does not display municipal data for cities, towns, etc.
- Full census and historical data requires professional subscription.
- Pro subscription available through several academic libraries, but few public libraries.

Quick overview of features

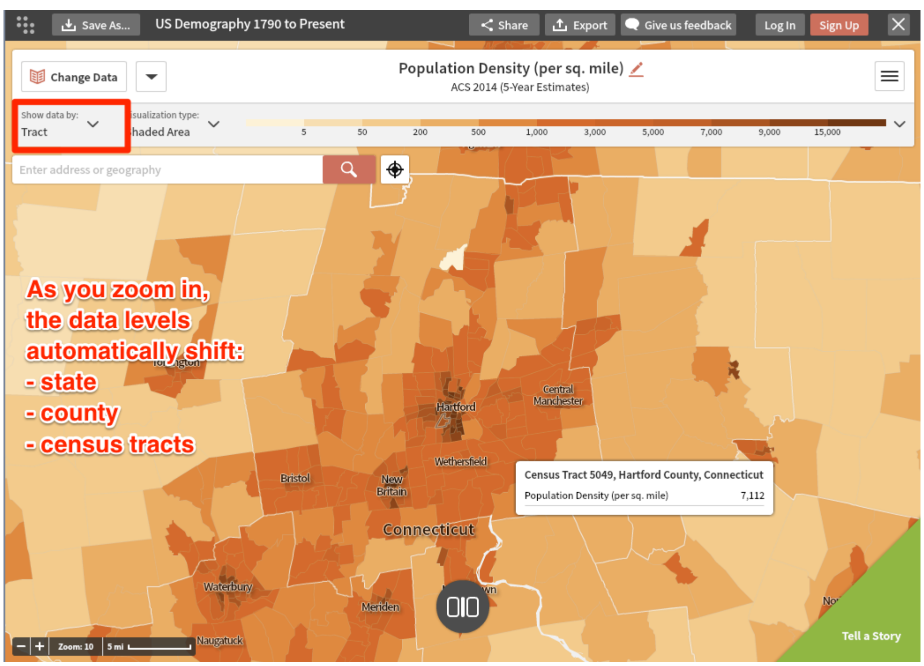
Start at the Social Explorer website http://socialexplorer.com and click on Maps. This tutorial demonstrates features available on the free edition.



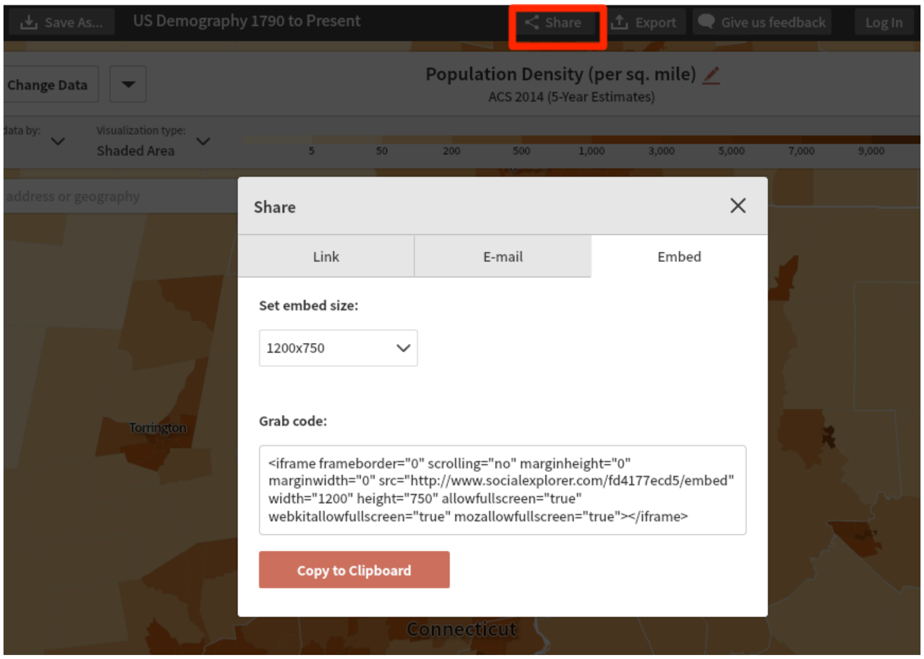
The default map view shows US population density, based on the American Community Survey (ACS) 5-year estimates. Click the Change Data button to explore other options.



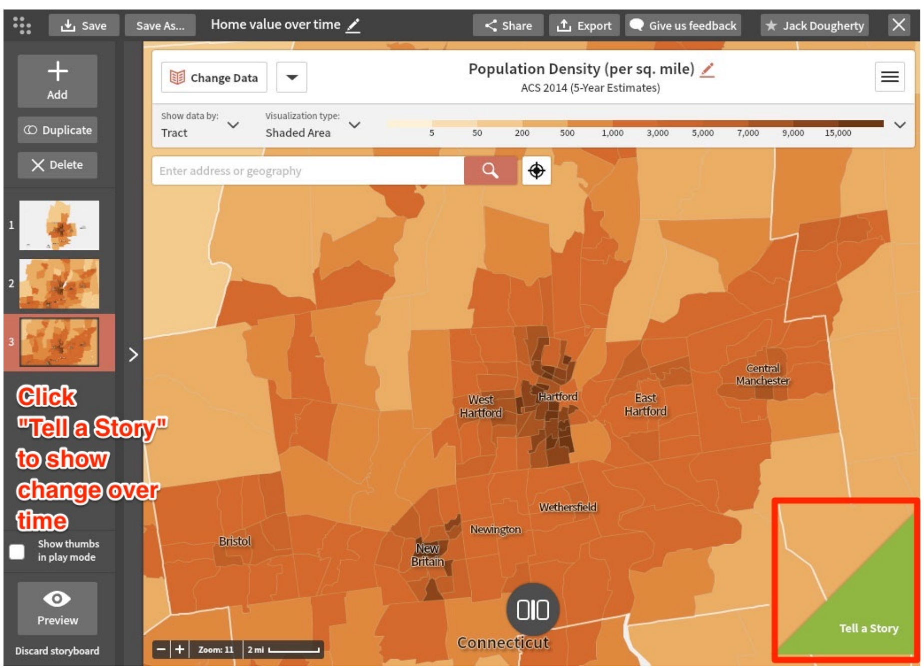
Geographic boundaries automatically change with the zoom level. As you zoom in, the data levels automatically shift from state, to county, to census tract.



Click the Share button to copy the link to your map, or the iframe code to embed it inside your own website.



Create a free account to save your online map views. Click the Tell a Story button, add a series of interactive map views, and show change over time.



TODO: Is this still true? All of the steps above can be done with the free version, but data is limited. Check if an academic library near you has a professional subscription.

Polygon Map with Tableau Public

Tableau Public is freely-available software that contains powerful tools to quickly create interactive polygon maps for common boundaries (such as US States, or World Nations.) If you need to create customized maps for less-common boundaries, see our chapter on Leaflet Maps with Google Sheets in this volume.

Important: Tableau Public is designed to publicly display your data, which makes this free tool very appropriate for educators, journalists, non-profit organizations, or other users who wish to openly share their map. If you desire a private tool to restrict your data, Tableau offers other tools that require payment.

See also the Tableau Public support page https://www.tableau.com/support/public for additional resources, including video tutorials.

1) Download and install the free Tableau Public tool, available for Mac or Windows, at https://public.tableau.com/en-us/s/download. Do not confuse with other Tableau products that require payment. Installation may require up to 5-10 minutes.

2) Click this link and Save to download to your computer: us-sample-data in CSV format. CSV means comma-separated values, a generic spreadsheet format that most tools can easily open. For help with downloading, see this short video tutorial.

3) Open the us-sample-data.csv file with any spreadsheet tool to view its contents.

4) Launch Tableau Public. In the Connect column of the first screen, click "Text file" to connect to the CSV file you downloaded above. (If you had an Excel file in .xlsx format, you would click that instead.) Navigate to select the us-sample.data.csv file on your computer.

5) At first, Tableau Public does **NOT** recognize the names of US areas, which initially appear simply as "text" values (with the "Abc" symbol). Click and hold down the mouse directly on the "Abc" symbol, and use the drop-down menu to convert to Geographic role > State/Province. A tiny globe symbol will appear to show that Tableau Public now recognizes this column as geographic data, which is essential in order to make a map.

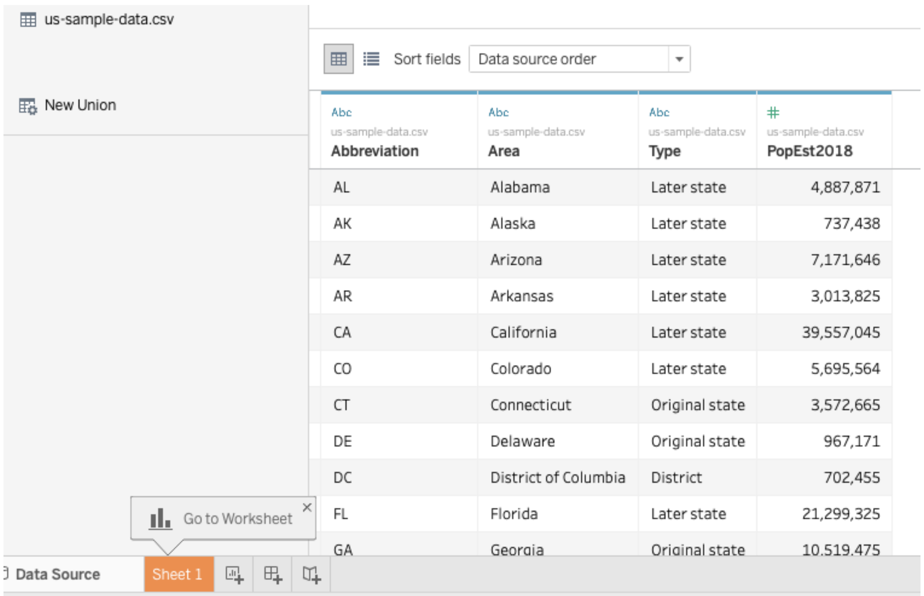


Figure 6.16: Column displayed as text data

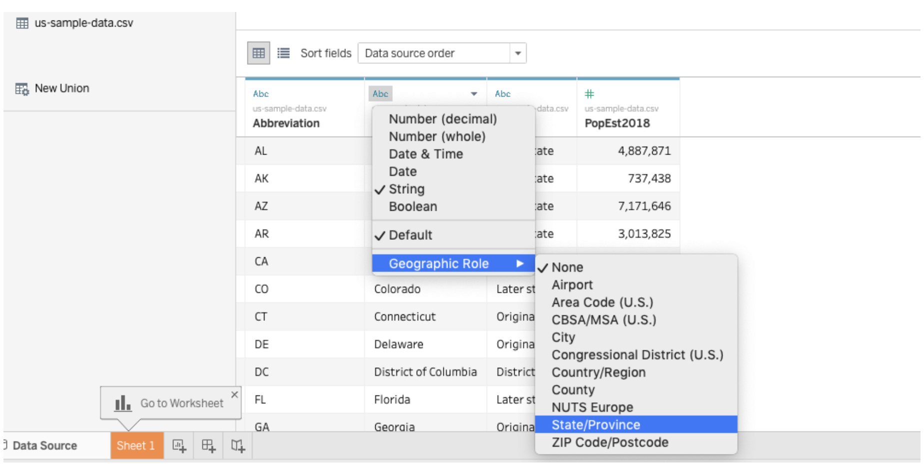


Figure 6.17: Column converted to geographic data

6) Go to the Worksheet view, by clicking on “Sheet 1” in the bottom-left corner. The goal is to build a polygon map, based on the dimensions and variables provided by Tableau Public.

Step A - Drag the “Area” dimension to the middle of the worksheet to create the geographic areas

Step B - In the “Marks” panel, change the drop-down menu from “Automatic” to “Map”

Step C - Drag the “Type” dimension into the “Color” box of the Marks panel to show polygon colors according to type

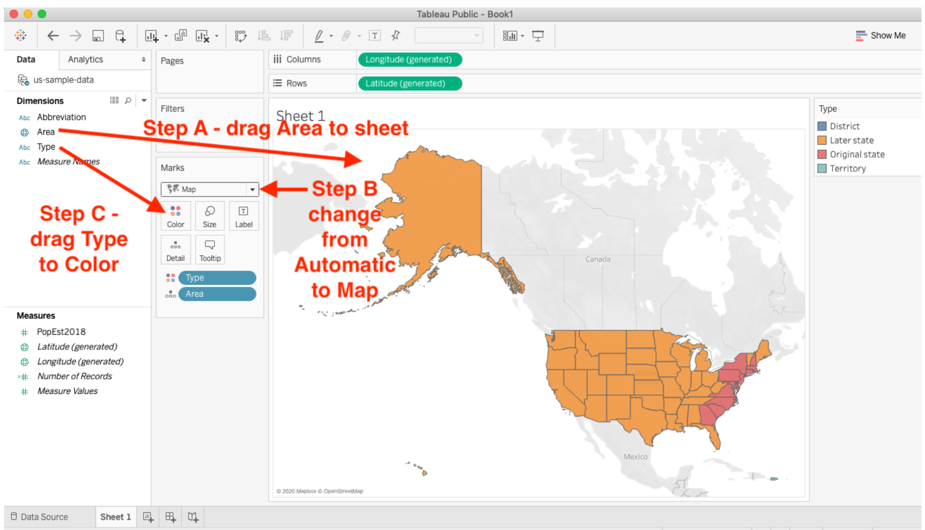


Figure 6.18: Steps A-B-C above

Optional: Add more items, such as “Abbreviation” dimension to “Label” box to display state abbreviations, or “Area” dimension to “tooltips” to display on mouseover.

7) To display your map online, click “Dashboard” tab in the bottom-left corner.

8) Drag “Sheet 1” (the default name of your map) into your dashboard. Also, drag the map legend from the corner into the lower body of the map (or choose other legend options).

9) To publish your map online, go to File > Save to Tableau Public As... This will require you to create a free Tableau Public Account.

10) Modify your final online product as desired, and see options to embed elsewhere on the web.

Chapter 7

Embed On Your Web

After you create a chart or map, how do display it inside your website as an *interactive* visualization? Our goal is not a static picture, but a live chart or map that users can explore. This is an important question for beginners, since data visualizations are not valuable unless you can control where and how your work appears. This chapter walks you through the key steps.

First, you need to own a website that supports iframe codes (which we'll explain below). If you do not have a website that supports this, then follow this quick tutorial to Create a simple web page with GitHub Pages. Even if you already have a website, still do this tutorial, because it introduces a tool used many times in this book.

Second, you need to copy or create an iframe code from your chart or map. An iframe is one line of HTML code with instructions on how to display a web page from a specific address (called a URL). A simple iframe looks like this:

```
<iframe src="https://handsondataviz.org/embed/index.html"></iframe>
```

No coding skills are necessary. See these easy-to-follow examples:

-Copy iframe from a Google Sheets chart -Convert a link into an iframe

Finally, you need to paste (or embed) the iframe code inside your website. Like a picture frame, an iframe allows you to display one web page (your data visualization) inside another web page (your personal website). But unlike a picture frame, where the image is static, an iframe makes content interactive, so visitors can explore the chart or map on your site, even though it may actually be hosted on an entirely different website. Go to this third tutorial, which combines the two steps above, called Embed Iframe in GitHub Pages.

See more tutorials in this chapter to copy iframes from other visualization tools (such as Tableau Public and embed them in other common websites (such as WordPress, etc.) ** TO DO: add more tutorials and links **

Enroll in our free online course **link to do**, which introduces these topics in the brief video below, and offers more exercises and opportunities to interact with instructors and other learners.

Create a Simple Web Page with GitHub Pages

Question: After you create an interactive chart or map, how do you embed the live version in a website that you control?

The full answer requires three steps:

- 1) Create a web page that supports iframe codes
- 2) Copy or create an iframe code from your visualization
- 3) Embed (or paste) the iframe code into your web page

This tutorial focuses on the **first step**. If you don't already have your own website, or if you are not sure whether your site supports iframe codes, then follow the steps below. We will create a simple web page with a free and friendly tool called GitHub http://github.com, and host it on the public web with the built-in GitHub Pages feature. For **steps 2 and 3**, see the Copy iframe from Google Sheets tutorial and the Embed iframe in GitHub Pages tutorial in this chapter.

Tool Review

GitHub http://github.com is a versatile tool that can be used to create simple web pages.

- Pros:
  - Free and easy-to-learn tool to edit and host simple pages on the public web.
  - All steps below can be completed in your web browser.
- Cons:
  - All work on GitHub is public by default. Private repositories (folders) require payment.
  - New users sometimes confuse the links for code repositories versus published web pages.

Video with step-by-step tutorial

1) Sign up for free GitHub account, then sign in, at http://github.com.
2) Create a new repository (also called a "project" or similar to a "folder").
3) Name your repository (or "repo"), and select Initialize with a README file. Optional steps: add a description and select a license.
4) Scroll down and click the green button to Create your repo, which will appear in a new browser tab, with this URL format:

```
https://github.com/YOUR-USERNAME/YOUR-REPO-NAME
```

5) In your GitHub repo, click on Settings, scroll down to GitHub Pages, select Master branch as your source, then Save. This publishes the code from your repo to the public web.

Hint: Do NOT select Theme Chooser for this exercise. It will create additional files that will interfere with displaying an iframe in your README.md file.

6) When the Settings page refreshes, scroll back down to GitHub Pages to see the new link to your published website, which will appear in this format:

```
https://YOUR-USERNAME.github.io/YOUR-REPO-NAME
```

7) Right-click and Copy the link to your published web site.
8) At the top of the page, click on the repo name to return to the main level.
9) Click the README.md file to open it in your browser, and click the pencil symbol to edit it.
10) Inside your README.md file, paste the link to your published web site, and type any text you wish to appear. The .md extension refers to Markdown, an easy-to-read computer language that GitHub Pages can process.
11) Scroll down and click the green Commit button to save your edits.
12) When your GitHub repo page refreshes, click on the new link to go to your published web site. **BE PATIENT!** Your new site may not appear instantly. Refresh the browser every 10 seconds. You may need to wait up to 1 minute for a new site to appear the first time, but later changes will be much faster.

Remember that GitHub Pages is designed to create simple web pages and sites. See other web publishing tools mentioned in this chapter to create more sophisticated web sites.

Copy an iframe code from a Google Sheets interactive chart

Question: After you create an interactive chart or map, how do you embed the live version in a website that you control?

The full answer requires three steps:

1. Create a web page that supports iframe codes
2. Copy the iframe code from your visualization
3. Embed (or paste) the iframe code into your web page

This tutorial focuses on the **second step**, and shows how to publish a Google Sheets interactive chart, and copy its iframe code. Details may differ for other visualization tools, but the general iframe concept will be similar to most cases. For **steps 1 and 3**, see the Create a Simple Web Page with GitHub Pages tutorial and the Embed iframe in GitHub Pages tutorial in this chapter.

Tutorial

1) Create a Google Sheets chart, which requires a free Google Drive account. Learn more in the Google Sheets Charts tutorial in this book.

2) Click the drop-down menu in the upper-right corner of the interactive chart and select Publish chart. Click OK on next screen.

3) Select the Embed tab, select the Interactive version, and click the blue Publish button. If you make changes to the chart, they will continue to be published to the web automatically, unless you click the Stop button or checkbox at the bottom.

4) Copy the iframe embed code.

No coding skills are necessary, but it helps to be code-curious. This iframe is a line of HTML code that contains these instructions:

- iframe tags to mark the beginning and end
- width and height: to display your chart in a second site, in pixels
- seamless frameborder: “0” means no border will appear around the chart in the second site
- scrolling: “no” means the chart will not include its own web scrolling feature

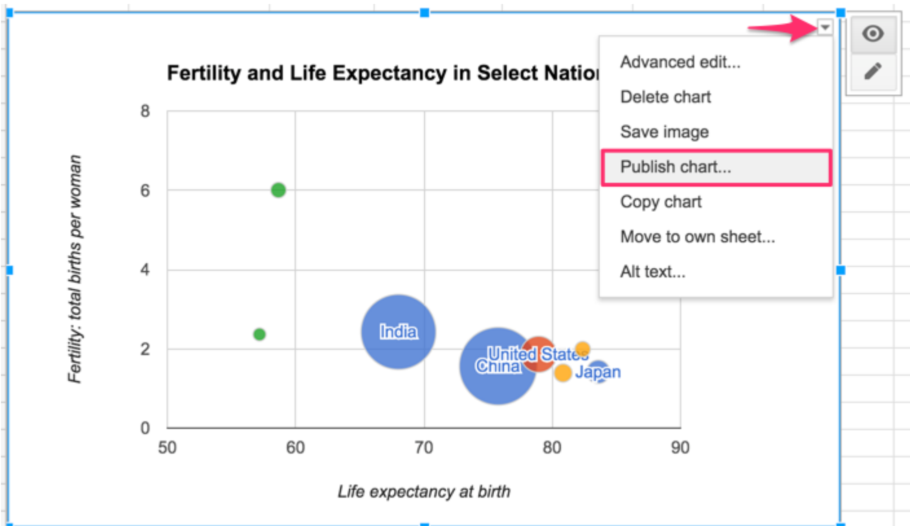


Figure 7.1: Screenshot: Drop-down menu to publish a Google Sheets chart

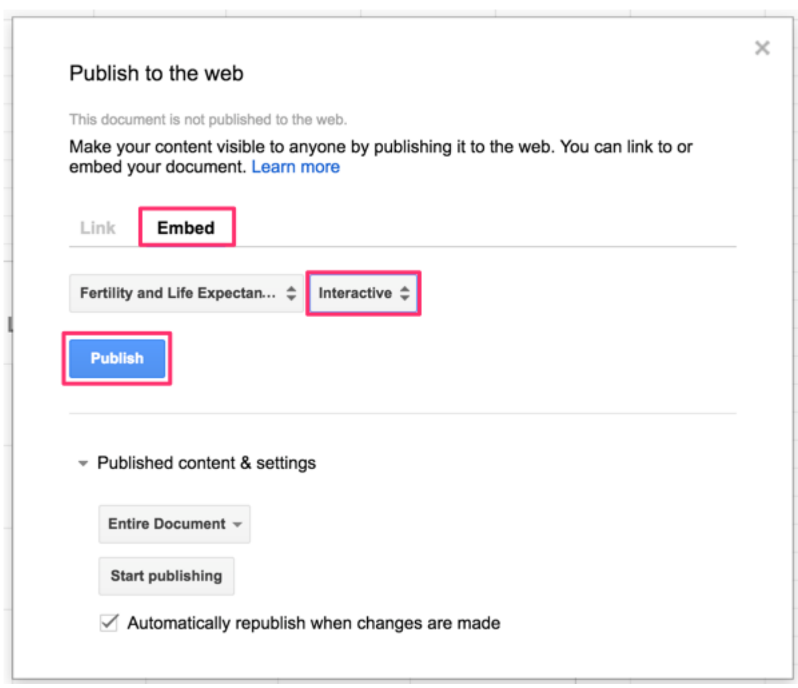


Figure 7.2: Screenshot: Publish to the web for a Google Sheets chart

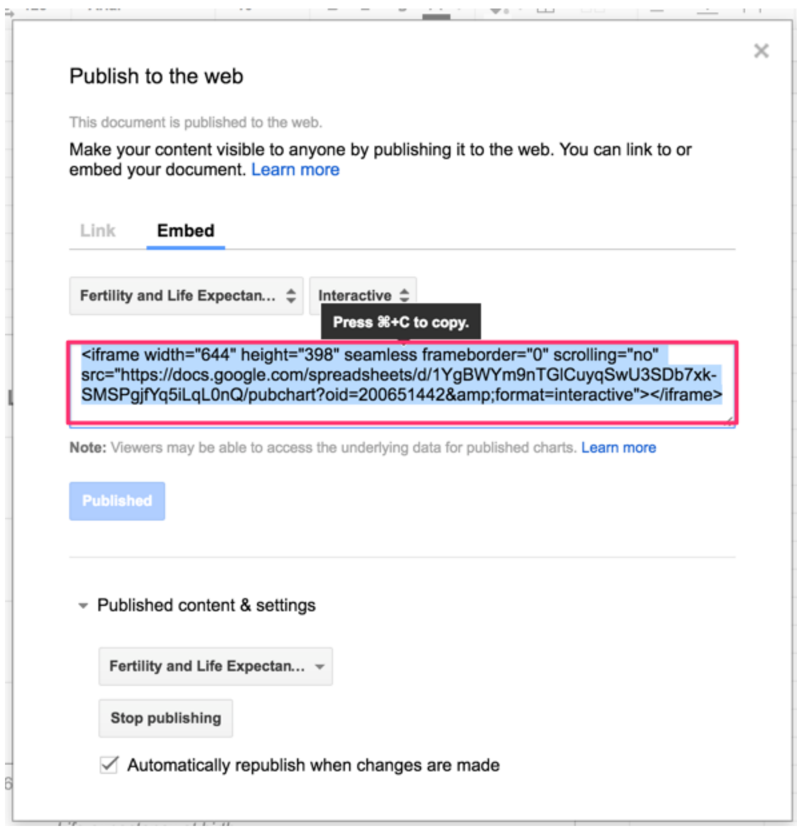


Figure 7.3: Screenshot: Copy the iframe code from a Google Sheets chart

- src: the web address (or URL) of the visualization to be displayed in the second site

See the next tutorial in this chapter, Embed iframe in GitHub Pages, to learn how to paste the iframe into a simple web page. Or see related tutorials in this chapter to embed an iframe in other common web sites.

Convert a Weblink into an Iframe

After you publish your data visualization to the web, how do you convert its weblink (or URL) into an iframe, to embed in your personal website?

The answer depends: did you publish your visualization as a code template on GitHub Pages? Or did you publish it using a drop-and-drag tool such as Google Sheets or Tableau Public?

Published with a code template on GitHub Pages

If you published your visualization from a code template (such as Leaflet or Chart.js) with GitHub Pages, follow these easy steps:

1) Copy the URL of your published visualization on GitHub, which will be in this format:

```
https://USERNAME.github.io/REPOSITORY
```

2) Add `iframe` tags to the beginning and end, insert `src=` and enclose the URL inside quotation marks, like this:

```
<iframe src="https://USERNAME.github.io/RESPOSITORY"></iframe>
```

3) Optional: Insert preferred width and height (in pixels by default, or percentages), like this:

```
<iframe src="https://USERNAME.github.io/RESPOSITORY" width="90%" height="400"></iframe>
```

4) Go to the appropriate tutorial to embed your iframe in your personal website:

- Embed an iframe in GitHub Pages
- Embed an iframe in WordPress.org

Published with Google Sheets or Tableau Public

Or, if you published your visualization using a drop-and-drag tool, see these tutorials:

- Copy an iframe code from a Google Sheets interactive chart
- Embed Tableau Public on your Website

Embed an Iframe in GitHub Pages

Question: After you create an interactive chart or map, how do you embed the live version in a website that you control?

Here's the full three-step answer that combines lessons from the Embed on the Web chapter introduction and the two previous tutorials:

1) First, create a web page that supports iframe embed codes. If you don't know what that means or don't yet have a personal website, go back to the previous tutorial, Create a Simple Web Page with GitHub Pages, or see the video and step-by-step instructions below.

2) Second, copy or create an iframe code from your data visualization. Go back to the previous tutorial, Copy an iframe code from a Google Sheets interactive chart, or see the video and step-by-step instructions below.

3) Third, embed (or paste) the iframe code into your website. The video and instructions below show how to paste an iframe from a Google Sheets interactive chart into a simple web page with GitHub Pages.

Try it

The goal is to embed the iframe code from a Google Sheets interactive chart, which resides on a Google web server, into your GitHub Pages web site. The result will be similar to the one below:

Video tutorial and step-by-step instructions

1) Sign up for free GitHub account, then sign in, at https://github.com.

2) Create a **new repository** (think of it as a folder that contains your project).

3) Name your repository (or "repo"), and select *Initialize this repository with a README.* Optional steps: add a description and select a license.

4) Scroll down and click the green button to Create your repo, which will appear in a new browser tab, with this URL format:

```
https://github.com/YOUR-USERNAME/YOUR-REPO-NAME
```

5) In your GitHub repo, click on Settings tab, scroll down to *GitHub Pages*, select **master branch** as your Source, then Save. This publishes the code from your repo to the public web.
6) When the Settings page refreshes, scroll back down to GitHub Pages to see the new link to your published website, which will appear in this format:

```
https://YOUR-USERNAME.github.io/YOUR-REPO-NAME
```

7) Right-click and Copy this link to your published web site.
8) At the top of the page, click on the repo name to return to the main level.
9) Click the README.md file to open it in your browser, and click the pencil symbol in the upper right corner to edit it.
10) Inside your README.md file, paste the link to your published web site, and type any text you wish to appear. The .md extension refers to Markdown, an easy-to-read markup language that GitHub Pages can process and display as HTML.
11) Go to a data visualization you have created, such as a Google Sheets chart, select Publish > Embed, and copy the iframe code. This line of HTML code displays the interactive visualization website inside your personal website.
12) Scroll down and click Commit to save your edits.
13) When your GitHub repo page refreshes, click on the new link to go to your published web site. **BE PATIENT!** Your new site may not appear instantly. Refresh the browser every 10 seconds. You may need to wait for a few minutes for a new site to appear the first time, but later changes will be much faster.

Important:

- A published README.md file will display an HTML iframe code, unless you add other HTML files (such as index.html) to your repository.

Remember that GitHub Pages is designed to create simple web pages and sites. See other web publishing tools mentioned in this chapter to create more sophisticated web sites.

Embed an Iframe on WordPress.org

TODO:

- rewrite this tutorial to merge the two versions (top and bottom)
- then update all links and check all `code` tags

To embed one web page (the data visualization) inside a second web page (the organization's website), we use a simple HTML code known as **iframe**. (Read more about the iframetag at W3Schools.)

The **general iframe concept** works across many data visualization tools and many websites: - Copy the embed code or URL from your dataviz website - Paste (and modify) the code as an iframe in your destination website

To embed your dataviz in a self-hosted Wordpress.org site, the [iframe plugin](http://wordpress.org/plugins/iframe/) must be installed and activated. This plugin allows authors to embed iframe codes inside posts/pages, in a modified "shortcode" format surrounded by square brackets. Without the plugin, self-hosted WordPress.org sites will usually "strip out" iframe codes for all users except the site administrator. **I have already installed and activated** the iframe plugin on my site, and the Dashboard view looks like this:

Note that most WordPress.com sites do NOT support an iframe embed code.

But details vary, so read and experiment with the examples that follow.

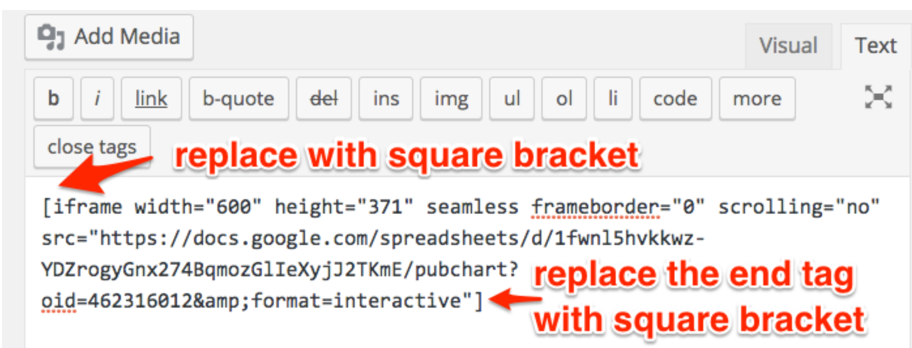
5) To embed the iframe in a WordPress.org site, the iframe plugin must be installed, as explained in the Embed with iframe on WordPress.org chapter. **TO DO** fix self-reference

6) Log into your Wordpress.org site and create a new post. In the editor window, switch from the Visual to the Text tab, which allows users to modify the code behind your post. Paste the iframe code from your interactive dataviz.

7) Initially, the code you pasted includes HTML iframe tags at the front <iframe... and the end ...></iframe>, which looks like this:

```
<iframe width="600" height="371" seamless frameborder="0" scrolling="no" src="https://docs.google
```

8) Modify the front end of the iframe code by replacing the less-than symbol (<) with a square opening bracket ([ ). Modify the back end by erasing the greater-than symbol (>) and the end tag ( ). Replace the back end with a square closing bracket ( ]).



Your modified code should look like this:

```
[iframe width="600" height="371" seamless frameborder="0" scrolling="no" src="https://docs.google
```

9) Click Preview or Publish/View Post to see how it appears on the web.

10) If desired, continue to modify the iframe code to improve the display of your dataviz on your website. For example, the initial code was 600 pixels wide (width="600"). To display the dataviz across the full width of your website, change this part of the code to 100% (width="100%").

The goal is to embed an interactive chart inside your website, so that users can explore the data. This tutorial displays a *very basic chart* to simplify the process, and the end result will appear like the one below. Try it.

Embed Tableau Public on your Website

Question: After learning how to create an interactive data visualization with Tableau Public in this book, how do I embed it on my website?

Answer: Tableau Public supports two embedding methods, and your choice depends on your type of website.

- A) Embed code: if you can paste directly into an HTML web page
- B) Convert Link to iframe: to paste into WordPress.org, Wix, SquareSpace, Weebly, and many other web platforms

Try it

Both methods produce an embedded visualization like the one below. Float your cursor over points to view data details.

A) Embed code method for HTML web pages

1) Use this method if you can paste HTML and JavaScript code directly into a website with HTML pages.
2) Go to the public web page of any Tableau Public visualization, such as this sample: https://public.tableau.com/profile/jackdougherty#!/vizhome/CTSchoolDistrictsbyIncomeandGradeLevels2009-13/Sheet1
3) Before you begin the embed process, click the upper-right Edit Details button to make any final modifications to the title or toolbar settings.
4) Click the bottom-right Share button, click inside the **Embed Code** field, and copy its contents. A typical embed code is a long string of HTML and JavaScript instructions to display the visualization.

5) Open an HTML page on your website and paste the embed code in the body section. Below is an example of a sample Tableau Public embed code pasted between the body tags of a simple HTML page.

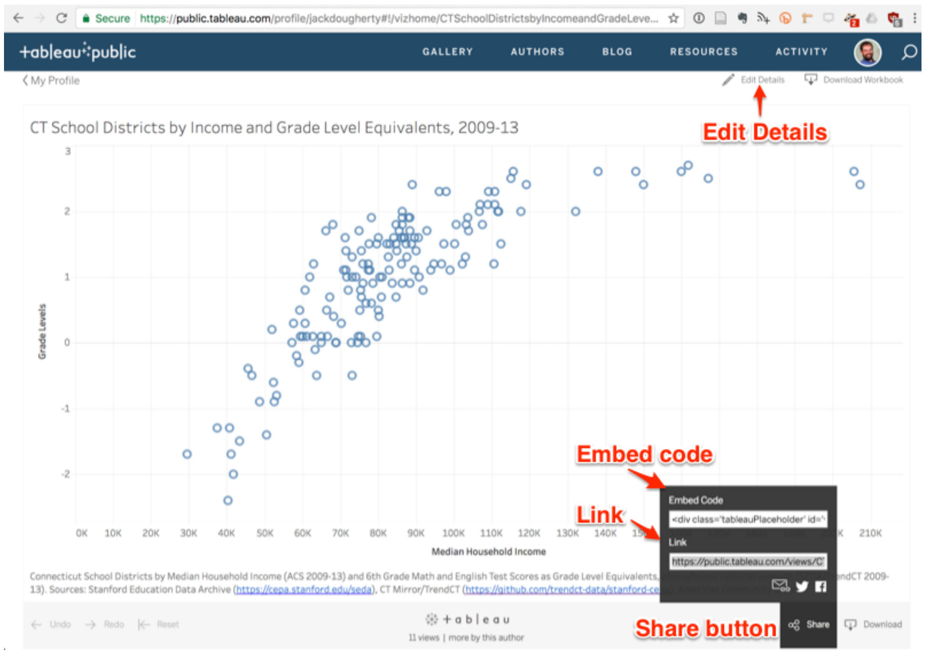


Figure 7.4: Screenshot: Edit and Share buttons in Tableau Public web page

```
<!DOCTYPE html>
<html>
<head>
  <title>sample web page</title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta charset="utf-8">
</head>
<body>
  <div class='tableauPlaceholder' id='viz1489158014225' style='position: relative'><noscript><a h
</body>
</html>
```

B) Convert Link to iframe method

1) Use this method if you need to paste an iframe into common web authoring platforms (such as WordPress.org, Squarespace, Wix, Weebly, etc.), since these platforms typically do not support HTML and JavaScript code pasted directly into content.

2) Go to the public web page of any Tableau Public visualization, such as this sample: https://public.tableau.com/profile/jackdougherty#!/

vizhome/CTSchoolDistrictsbyIncomeandGradeLevels2009-13/Sheet1

3) Before you begin the embed process, click the upper-right Edit Details button to make any final modifications to the title or toolbar settings.

4) Click the bottom-right Share button, click inside the **Link** field (NOT the Embed Code field), and copy its contents.

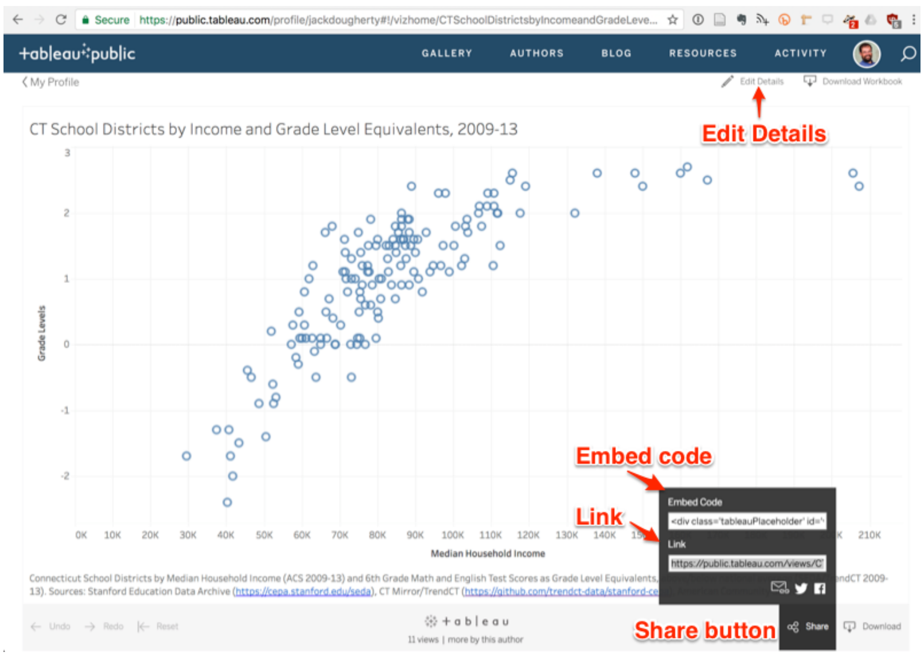


Figure 7.5: Screenshot: Edit and Share buttons in Tableau Public web page

5) A typical link will look similar to this example (scroll to right to see all):

```
https://public.tableau.com/views/CTSchoolDistrictsbyIncomeandGradeLevels2009-13/Sheet1
```

6) We need to edit the link to convert it into an iframe format. First, delete any code that appears after the question mark, to make it look like this (scroll to right to see all):

```
https://public.tableau.com/views/CTSchoolDistrictsbyIncomeandGradeLevels2009-13/Sheet1
```

7) Add this snippet of code to the end, to replace what you deleted above:

```
:showVizHome=no&:embed=true
```

8) Now your edited link should look similar to this (scroll to right to see all):

```
https://public.tableau.com/views/CTSchoolDistrictsbyIncomeandGradeLevels2009-13/Sheet1?:showVizHo
```

9) Enclose the link inside an iframe source tag `src=` with quotes, to make it look similar to this (scroll to right to see all):

```
src="https://public.tableau.com/views/CTSchoolDistrictsbyIncomeandGradeLevels2009-13/Sheet1?:show
```

10) Add iframe tags for `width` and `height` in percentages or pixels (default), to make it look similar to this (scroll to right to see all):

```
src="https://public.tableau.com/views/CTSchoolDistrictsbyIncomeandGradeLevels2009-13/Sheet1?:show
```

Hint: Insert 90% width, rather than 100, to help readers easily scroll down your web page

11) Add iframe tags at the beginning and end, to make it look similar to this (scroll to right to see all):

```
<iframe src="https://public.tableau.com/views/CTSchoolDistrictsbyIncomeandGradeLevels2009-13/Shee
```

Exceptions to the last step above. As described in the Embed iframe on WordPress chapter in this book, in a self-hosted WordPress.org site, with the iframe plugin, insert iframe brackets rather than HTML tags to make a shortcode like this (scroll to right to see all):

```
[iframe src="https://public.tableau.com/views/CTSchoolDistrictsbyIncomeandGradeLevels2009-13/Shee
```

Learn more

Embedding Tableau Public Views in iframe, Tableau Support page http://kb.tableau.com/articles/howto/embedding-tableau-public-views-in-iframes

Chapter 8

Modify and Host Code with GitHub

In the first half of this book, we explored free web services that offer easy drag-and-drop tools to create interactive charts and maps, such as Google Sheets, Google My Maps, Carto, and Tableau Public. But these web services have limited options for designing and customizing your visualizations, and also make you dependent on their web servers to host your work. In this second half of the book, we'll explore how to copy, edit, and host code templates, meaning pre-written software instructions to create visualizations. With templates, no prior coding skills are necessary. You will learn how to make simple edits to insert your data, customize its appearance, and display it on the web on a site you control.

Enroll in our free online course, which introduces these topics in the brief video below, and offers more exercises and opportunities to interact with instructors and other learners.

Video overview

Tool Review

GitHub (http://github.com) is a versatile tool to share, edit, and host simple code templates on the public web. Requires a free account. Although advanced coders use more powerful command-line versions of this tool, this introduction demonstrates all of the basic steps using GitHub in the web browser.

- Pros:
  - Free and easy-to-learn tool that beginners can use in the web browser.

  - Popular tool to share, copy, and edit open-source code repositories (project folders).
  - Host simple code (such as HTML/CSS/JavaScript) on the live web with GitHub Pages.
  - Built-in support to quickly display open-data formats: CSV tables and GeoJSON geography.
  - Easy to migrate code repositories to a different web server.
- Cons:
  - By default, all work on GitHub is public. Private repositories require payment.
  - New users often confuse web addresses for code repository versus published web page.

In this chapter, you will learn how to:

- Fork and edit a simple Leaflet map with GitHub
- Fork and edit a Highcharts scatter chart with GitHub
- Create a new repository and upload code with GitHub
- Choose an open-source code license
- Pull request to merge changes on GitHub
- Work more efficiently with Atom editor and GitHub Desktop
- Fix Common Code and GitHub Errors

Fork and Edit a Simple Leaflet Map with GitHub

This tutorial introduces the **basic steps** of working with code templates, using a simple Leaflet map code (http://leafletjs.com) and GitHub in your browser (http://github.com). You will learn how to:

- A) Fork (copy) Leaflet template to your GitHub account
- B) Publish your live map to public web with GitHub Pages
- C) Modify your map title and add layer controls
- D) Geocode addresses in a Google Sheet and upload points from data.csv

Code templates help us to move beyond the limits of drag-and-drop web mapping services (such as Google MyMaps) and to create more customized visualizations on a web server that you control. Before you begin, learn the broad concepts in the chapter introduction Modify and Host Code Templates with GitHub. For more advanced examples, see the Leaflet Map Templates chapter in this book. If you have problems with this tutorial, go to the Fix Common GitHub and Code Errors chapter in this book.

Try it

You will begin this tutorial with a simple interactive map that includes one pop-up point:

By the end of this tutorial, you will learn how to modify the map, then geocode and upload more data points:

Video with step-by-step tutorial

A) Fork (copy) Leaflet template to your GitHub account

Before you begin, sign up for a free GitHub account: http://github.com

1) Right-click to open this GitHub code template in a new tab: https://github.com//leaflet-map-simple
2) In the upper-right corner of the code template, sign in to your free GitHub account
3) In the upper-right corner, click Fork to copy the template (also called a code repository, or repo) into your GitHub account. The web address (URL) of the new copy in your account will follow this format:

```
https://github.com/USERNAME/REPOSITORY
```

Reminder: You can only fork a GitHub repo **one time**. If needed, see how to make a second copy in the Create a New Repo in GitHub chapter in this book.

B) Publish your live map to public web with GitHub Pages

4) In your new copy of the code repo, click on Settings, scroll down to the GitHub Pages area, select Master, and Save. This publishes your code template to a live map on a public website that you control.
5) Scroll down to GitHub Pages section again, to select and copy the link to your published web site, which will follow this format:

```
https://USERNAME.github.io/REPOSITORY
```

6) Scroll up to the top, and click on your repo name to go back to its main page.
7) At the top level of your repo main page, click on README.md, and click the pencil icon to edit this file, written in easy-to-read Markdown code.

8) Delete the link to the current live site, and paste in the link to your site. Scroll down and Commit to save your edits.

9) On your repo main page, right-click on the link to your published site to open in a new tab. **Be patient** during busy periods, because your website may take up to 1 minute to appear the first time.

C) Modify your map title and add layer controls

10) Go back to your browser tab for your code repo. Click on the index.html file (which contains the map code), and click the pencil icon to edit it.

11) Explore the map code, which contains HTML, CSS, and JavaScript. Look for sections that begin with "EDIT" for items that you can easily change. Scroll down to Commit your changes.

12) Go to your live website browser tab and refresh the page to view your edits. **Be patient** during busy periods, when some edits may take up to 1 minute to appear.

13) To change your map title in the index.html file, click the pencil symbol (to edit) and go to lines 23-25. Replace "EDIT your map title" with your new title:

```
<!-- Display the map and title with HTML division tags  -->
<div id="map-title">EDIT your map title</div>
<div id="map"></div>
```

14) To change your initial map zoom level, edit the index.html file and go to line 33. The zoom range for this map is from 1 (max zoom out) to 18 (max zoom in).

```
// Set up initial map center and zoom level
var map = L.map('map', {
  center: [41.77, -72.69], // EDIT latitude, longitude to re-center map
  zoom: 12,  // EDIT from 1 to 18 -- decrease to zoom out, increase to zoom in
  scrollWheelZoom: false
});
```

15) To change the default basemap, edit lines 46 and 52 to delete ".addTo(map)" from the Carto light layer, then add it to the Stamen colored terrain layer. DO NOT erase the semicolons!

Your original code looks like this (scroll to right to see all):

```
/* Carto light-gray basemap tiles with labels */
  var light = L.tileLayer('https://cartodb-basemaps-{s}.global.ssl.fastly.net/light_all/{z}/{x}/{
    attribution: '&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>, &cop
  }).addTo(map); // EDIT - insert or remove ".addTo(map)" before last semicolon to display by def
  // controlLayers.addBaseLayer(light, 'Carto Light basemap');
  /* Stamen colored terrain basemap tiles with labels */
  var terrain = L.tileLayer('https://stamen-tiles.a.ssl.fastly.net/terrain/{z}/{x}/{y}.png', {
    attribution: 'Map tiles by <a href="http://stamen.com">Stamen Design</a>, under <a href="http
  }); // EDIT - insert or remove ".addTo(map)" before last semicolon to display by default
  // controlLayers.addBaseLayer(terrain, 'Stamen Terrain basemap');
```

After you edit the code, it should look like this (scroll to right to see all):

```
/* Carto light-gray basemap tiles with labels */
var light = L.tileLayer('https://cartodb-basemaps-{s}.global.ssl.fastly.net/light_all/{z}/{x}/{y}
  attribution: '&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>, &copy;
}); // EDIT - insert or remove ".addTo(map)" before last semicolon to display by default
// controlLayers.addBaseLayer(light, 'Carto Light basemap');
/* Stamen colored terrain basemap tiles with labels */
var terrain = L.tileLayer('https://stamen-tiles.a.ssl.fastly.net/terrain/{z}/{x}/{y}.png', {
  attribution: 'Map tiles by <a href="http://stamen.com">Stamen Design</a>, under <a href="http:/
}).addTo(map); // EDIT - insert or remove ".addTo(map)" before last semicolon to display by defau
// controlLayers.addBaseLayer(terrain, 'Stamen Terrain basemap');
```

16) To add a control panel that turns on/off map layers, delete the code comment symbols (//) that appear in front of lines 38-41, 47, and 53 to activate these sections. When you remove code comments in GitHub, the color changes from gray text (inactive code) to colored text (active code). After you remove the code comments, your file should look like this (scroll to right to see all):

```
/* Control panel to display map layers */
 var controlLayers = L.control.layers( null, null, {
  position: "topright",
  collapsed: false
 }).addTo(map);

/* Carto light-gray basemap tiles with labels */
var light = L.tileLayer('https://cartodb-basemaps-{s}.global.ssl.fastly.net/light_all/{z}/{x}/{y}
  attribution: '&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>, &copy;
}); // EDIT - insert or remove ".addTo(map)" before last semicolon to display by default
 controlLayers.addBaseLayer(light, 'Carto Light basemap');
/* Stamen colored terrain basemap tiles with labels */
var terrain = L.tileLayer('https://stamen-tiles.a.ssl.fastly.net/terrain/{z}/{x}/{y}.png', {
```

```
  attribution: 'Map tiles by <a href="http://stamen.com">Stamen Design</a>, under <a h
}).addTo(map); // EDIT - insert or remove ".addTo(map)" before last semicolon to displ
 controlLayers.addBaseLayer(terrain, 'Stamen Terrain basemap');
```

17) To change one point on the map, you could edit the latitude and longitude coordinates of the single marker in lines 55-57. To find coordinates for any location and to learn more, go to http://www.latlong.net

```
/* Display a blue point marker with pop-up text */
L.marker([41.77, -72.69]).addTo(map) // EDIT latitude, longitude to re-position marker
.bindPopup("Insert pop-up text here"); // EDIT pop-up text message
```

But a better way to display several points is to remove the code comment symbols (//) in front of lines 60-69 to activate this section of code, which pulls map points from the data.csv file in your GitHub repository. After your edits, this section should look like this (scroll right to see all):

```
/* Upload Latitude/Longitude markers from data.csv file, show Title in pop-up, and ove
 var customLayer = L.geoJson(null, {
  onEachFeature: function(feature, layer) {
    layer.bindPopup(feature.properties.Title);
  }
 });
 var runLayer = omnivore.csv('data.csv', null, customLayer)
 .on('ready', function() {
  map.fitBounds(runLayer.getBounds());
 }).addTo(map);
 controlLayers.addOverlay(customLayer, 'Markers from data.csv');
```

D) Geocode addresses in Google Sheet and upload points from data.csv

18) A better way to display multiple points on your map is to prepare and upload a new data.csv file to your GitHub repository. First, right-click to open this Google Sheets template in a new tab: Leaflet Maps Simple data points with Geocoder

19) Since this sheet is view-only, you cannot edit it. Instead, sign in to your Google account in the upper-right corner.

20) Go to File > Make a Copy, which will save a duplicate version to your Google Drive, which you can edit.

21) In your copy of the Google Sheet, select any cells and press Delete on your keyboard to erase contents. Type new titles and addresses into columns A and B.

22) To geocode your new addresses (which means converting them into latitude and longitude coordinates), select all of the contents across 6 columns, from Address (B) to Source (G).

23) Go to the Geocoder menu that appears in this special Google Sheet template, and select any service, such as US Census (for US addresses) or Google Maps. The first time you run the geocoder, the script will ask for permission.

24) After you have geocoded your addresses, go to File > Download As > Comma-separated values (.CSV format) to save the file to your computer.

25) In your computer, right-click the downloaded file to rename it to: data.csv

26) In your GitHub repository, click Upload Files, then drag-and-drop your new data.csv file, and Commit to upload it. Go to your live map browser tab and refresh to view changes. **Be patient* during busy periods, when some edits may take up to 1 minute to appear.**

Learn more

- To solve problems, see Fix Common GitHub and Code Errors chapter in this book.
- See more advanced Leaflet Map Templates in this book
- About Leaflet https://leafletjs.com
- GitHub Pages features and tutorial, https://pages.github.com

Fork and Edit a Highcharts Scatter Chart with GitHub

TODO: Decide whether to keep or replace with Chart.js

This tutorial introduces the **basic steps** of working with code templates, using a simple Highcharts scatter chart code (http://highcharts.com) and GitHub in your browser (http://github.com). You will learn how to:

- A) Fork (copy) the Highcharts template to your GitHub account
- B) Publish your live chart to the public web with GitHub Pages
- C) Modify the chart title, subtitle, and axis labels

- D) Upload new data points from a comma-separated values (.csv) spreadsheet

Code templates help us to move beyond the limits of drag-and-drop web tools (such as Google Sheets and Tableau Public) and to create more customized visualizations on a web server that you control. Before you begin, learn the broad concepts in the chapter introduction Modify and Host Code Templates with GitHub. For more advanced examples, see the Highcharts Templates chapter in this book. If you have problems with this tutorial, go to the Fix Common GitHub and Code Errors chapter in this book.

Try it

You will begin this tutorial with a basic chart template that includes only 7 points. Right-click to open full-size chart in new tab.

By the end of this tutorial, you will learn how to modify the chart and add a new CSV spreadsheet with over 160 points. Right-click to open full-size chart in new tab.

Video with step-by-step tutorial

A) Fork (copy) the Highcharts template to your GitHub account

Before you begin, sign up for a free GitHub account: http://github.com

1) Right-click to open this GitHub code template in a new tab: https://github.com//highcharts-scatter-csv

2) In the upper-right corner of the code template, sign in to your free GitHub account

3) In the upper-right corner, click Fork to copy the template (also called a code repository, or repo) into your GitHub account. The web address (URL) of the new copy in your account will follow this format:

```
https://github.com/USERNAME/REPOSITORY
```

Reminder: You can only fork a GitHub repo **one time**. If needed, see how to make a second copy in the Create a New Repo in GitHub chapter in this book.

B) Publish your live chart to the web with GitHub Pages

4) In your new copy of the code repo, click on Settings, scroll down to the GitHub Pages area, select Master, and Save. This publishes your code template to a live map on a public website that you control.
5) Scroll down to GitHub Pages section again, to select and copy the link to your published web site, which will follow this format:

```
https://USERNAME.github.io/REPOSITORY
```

6) Scroll up to the top, and click on your repo name to go back to its main page.
7) At the top level of your repo main page, click on README.md, and click the pencil icon to edit this file, written in easy-to-read Markdown code.
8) Delete the existing link to the live site, and paste in the link to your site. Scroll down and Commit to save your edits.
9) On your repo main page, right-click on the link to your published site to open in a new tab. **Be patient** during busy periods, when your website may take up to 1 minute to appear the first time.

C) Modify the chart title, subtitle, and axis labels

10) Go back to your browser tab for your code repo. Click on the index.html file (which contains the chart code), and click the pencil icon to edit it.
11) Explore the chart code, which contains HTML, CSS, and JavaScript. Look for code comments that begin with "EDIT" for sections that you can easily change, such as title, subtitle, x-axis and y-axis labels, and tooltip data labels. Scroll down to Commit your changes.
12) Go to your live website browser tab and refresh the page to view your edits. **Be patient** during busy periods, when some edits may take up to 1 minute to appear.

D) Upload new data points from a .CSV spreadsheet

13) Go to your GitHub code repository tab and click to view the file named: data-scatter.csv
14) GitHub automatically opens CSV files. Although it's possible to edit the file inside GitHub, let's upload a larger data file with the same name. Click this link and Save to download to your computer: data-scatter in CSV format.

15) In your GitHub code repo, click Upload Files, and drag the new data-scatter.csv into the folder, and Commit changes to replace the existing file with the same name.

16) In your GitHub repo, click the new data-scatter.csv file to inspect the changes. Then go to your live website tab and refresh to see the updated scatter chart. ** Be patient** during busy periods, when changes make take up to 1 minute to appear.

Learn more

- To solve problems, see the Fix Common GitHub and Code Errors chapter in this book.
- See more Highcharts Templates in this book
- Highcharts Demos http://highcharts.com/demo and Highcharts Docs http://www.highcharts.com/docs
- GitHub Pages features and tutorial, https://pages.github.com

Create a New Repo and Upload Code with GitHub

Question: If I already forked one copy of a GitHub code repository, GitHub will not allow me to fork it a second time. So how do I make a second copy of a repo?

Answer: GitHub has a "one-fork" rule for good reasons, but here's a simple way for beginners to work around it, using only your web browser and any computer (such as Mac, Windows, or Chromebook).

- Create a brand-new repository on GitHub in your browser
- Download an existing code repository and unzip the folder
- Upload the contents of that folder to your new repository and Commit Changes

Video with step-by-step tutorial

1) Follow these steps if you have already forked a GitHub repository and wish to make a second copy of it. For example, imagine that you have already forked a copy of the Leaflet Maps with Google Sheets repository from https://github.com//leaflet-maps-with-google-sheets **TO DO change repo address**

2) If you try to "fork" it again, GitHub will simply send you back to the first forked copy you already made. Clicking the "fork" button a second time is useless here.
3) Instead, go to your GitHub account and Create a New Repository. Give it a different name, and click the box to create a README.md file, then scroll down to click the Create button.
4) Go to the original repository where you wish to make a second copy, and click the Clone or Download button, and Download a zipped (compressed) file to your computer.
5) In your computer downloads folder, unzip the compressed file, typically by double-clicking it.
6) Go to the top level of your brand-new GitHub repository, and click the Upload Files button. Drag-and-drop all of the contents of the code repo you downloaded, EXCEPT the README.md file, because you have already created a new one. Click the Commit Changes button and be patient. During busy periods, a large upload may take 1 minute or more for GitHub to process.
7) When the upload is done, inspect the contents that you copied into your brand-new repository. To publish your new repo to the live web, go to Settings > GitHub Pages > select Master branch > Save. Then copy the link to your published live site and paste into your README.md file for future reference. If you need to review these last steps, see Part B: Publish section of the Fork and Edit a Leaflet Map chapter in this book.

Choose an Open-Source Code License

Whether you create a new code repository on GitHub, or fork a copy of someone else's code, you should understand the basic concepts of a software license. Developers have the option to add a `LICENSE` file to their GitHub repo, which explains what other people can (or cannot) do with their code.

On GitHub, free repos are publicly viewable and forkable by other users, so the platform encourages the use of open-source licenses. One example that is commonly used for code templates linked to this book is the MIT License: it allows anyone to copy, modify, and redistribute the code, as long as they credit the author(s) and do not hold them liable. Learn more about different types of open-source code licenses at http://choosealicense.com

Learn more

Licensing a Repository, GitHub Help page, https://help.github.com/articles/licensing-a-repository/

Pull Request to Merge Changes on GitHub

TODO: REWRITE this out-of-date page to focus solely on pull requests, branches, and merge changes

Sign up for a free GitHub account, a free multi-purpose tool that allows you to:

- View and fork a copy of open-source code from other users
- Make simple edits to your code directly in the browser
- Share your code and receive or suggest revisions to others
- Host a live version of your web code with GitHub Pages

GitHub is free if you publicly share your work. Private accounts require a subscription.

This chapter shows the basic steps to use GitHub **entirely in your browser**, which works well for new users on nearly any computer (Mac, Windows, Chromebook, etc.) Intermediate users will want to read the next chapter with supplemental tools: GitHub Desktop and Atom editor. Advanced users may prefer to use GitHub command-line instructions, which are beyond the scope of this book.

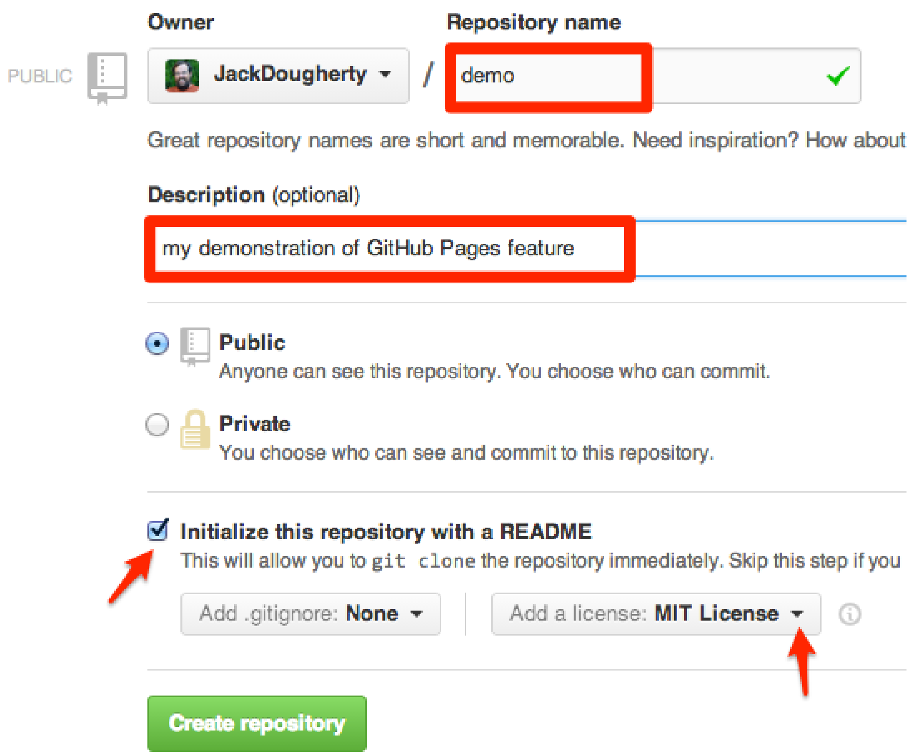
TO DO: REWRITE directions below to point users to GitHub Desktop and Atom Editor for editing on personal computer

Newcomers can host their code on GitHub, and publish to the web using the GitHub Pages feature, by following step-by-step instructions or this YouTube video screencast.

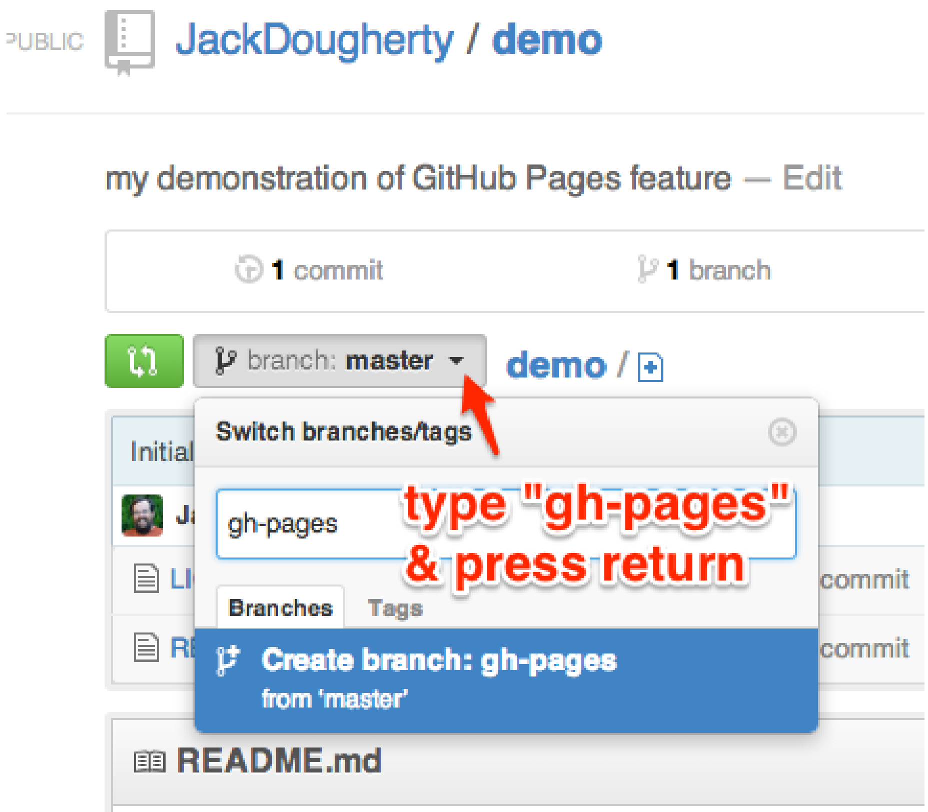
This basic tutorial demonstrates how to work with GitHub entirely through the browser. More advanced GitHub users may download other free tools (such as GitHub for Mac or GitHub for Windows or use other methods (such as the terminal command line) to work more efficiently.

1) Inside your free GitHub account, create a new repository (also known as a repo) to host your project's code (such as an index.html file and more).

2) Enter a repository name and description and check the box to automatically add a README file. If desired, select an open-source license (such as MIT), and click the Create Repository button.

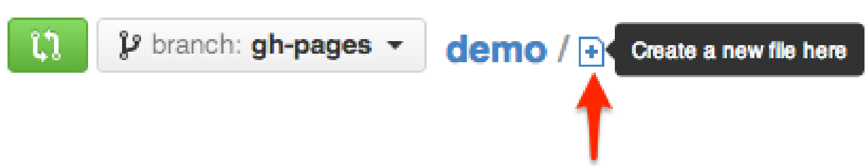


3) Your new repository automatically starts opens the "master" branch. Use the drop-down menu to create a new branch, and name it "gh-pages" (which is short for GitHub Pages), and press enter or return on your keyboard.



In this tutorial, we do all of our editing and testing work in the gh-pages branch, which automatically appears on the public web. When we're done, we will pull a copy (or sync) our completed work to the master branch for safekeeping and open sharing.

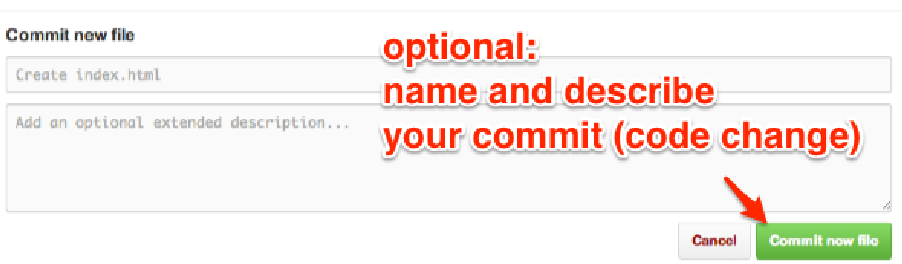
4) To add ONE NEW FILE to the gh-pages branch, click the + button next to the repository name and enter the file name. For example, index.html is the default file name for most web projects.



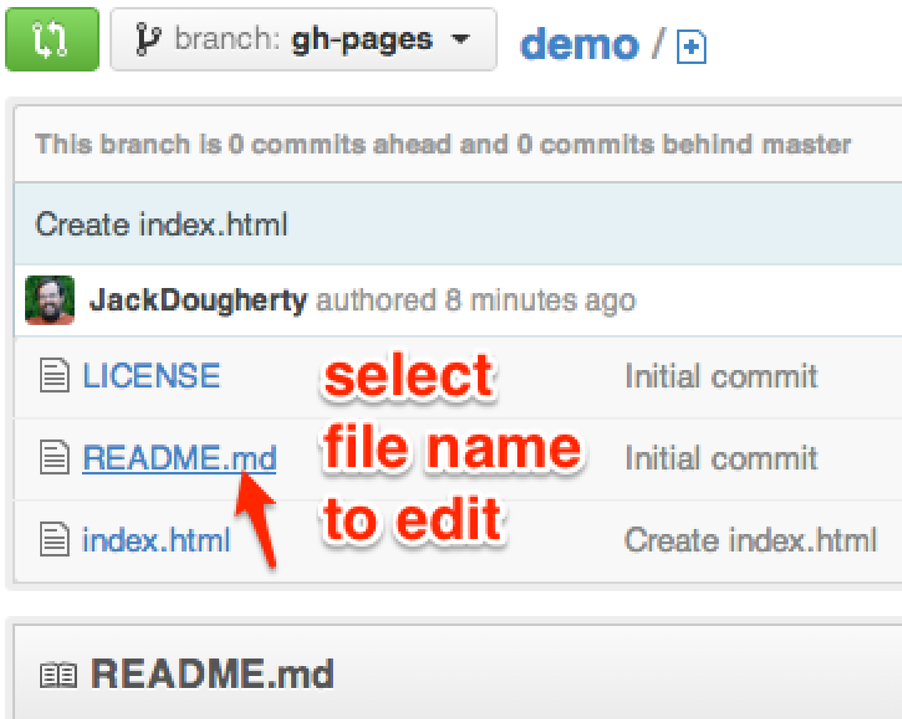
5) Select the "soft wrap" option (which makes long code strings more readable), and paste code into the editor. In this example, I pasted HTML

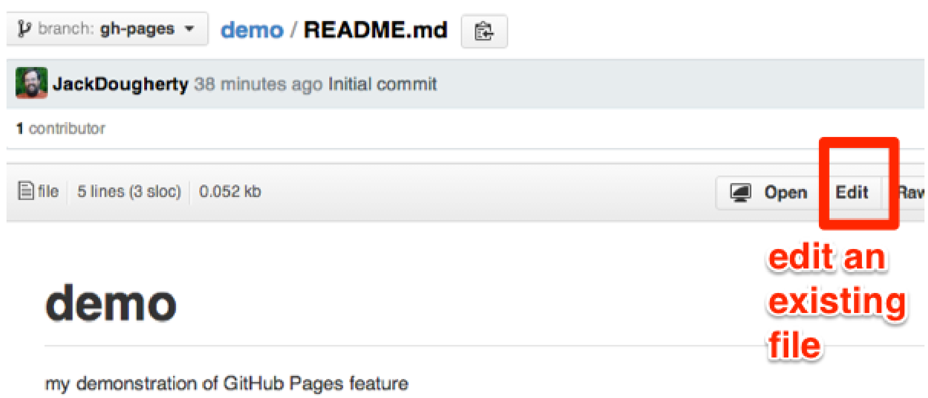
code that was generated by publishing an interactive chart from a Google Spreadsheet.

6) At the bottom, select the “commit” button (which means you are making a code change). Optionally, name and describe your commit, if you wish to track changes to your work.



7) Edit an existing file in the gh-pages branch by selecting its title. For example, select the README filename (which should have been automatically generated when you created the repository and the gh-pages branch). On the next screen, select the Edit button.

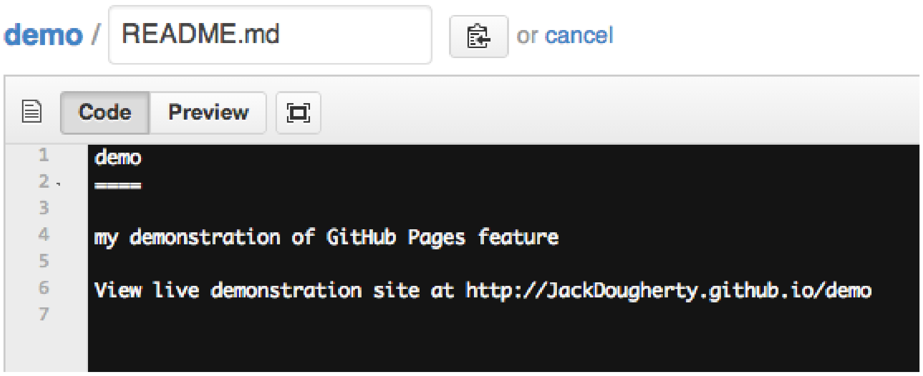




8) When editing the README file, type a link to the live web version of this repository, so that visitors may easily click to view it. The GitHub generic public web address is a combination of your username and ".github.io/" and your repository name, like this: `http://USERNAME.github.io/REPOSITORYNAME`

In my README file, I typically write it this way so that visitors know to click the link: View live demonstration site at *TO DO: insert link*

Commit your change to the README file in the gh-pages branch by pressing the green button at the bottom of the editor page.

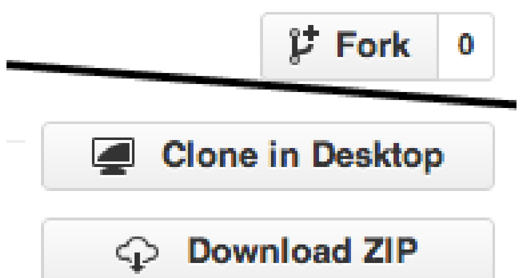


Hint: If you named your file "index.html" then you don't need to add anything else to the web address, because the site will automatically point to this default file. But if you entered a different file name, such as "sample.html", then you need to add it to the web address in this way: `http://Username.github.io/Repositoryname/sample.html`

9) Select the repository name to go to the upper-most file in the gh-pages branch.

10) To view your live index.html code on the public web, click the link you created in your README file. (Hint: use the right-click feature to open in a new tab/window). Important: The very first time you create a gh-pages branch, it may take up to 10 minutes for its content to appear on the open web. Afterwards, when you add or edit files in this branch, they should appear nearly instantly on the web, though you may need to refresh your browser to view any changes you have made.

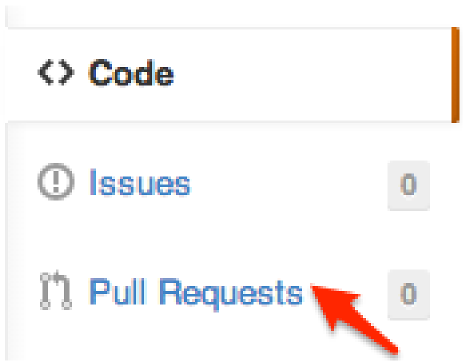
11) To add MULTIPLE FILES to the gh-pages branch, you could copy and paste each one individually as shown above, or choose one of these labor-saving options:



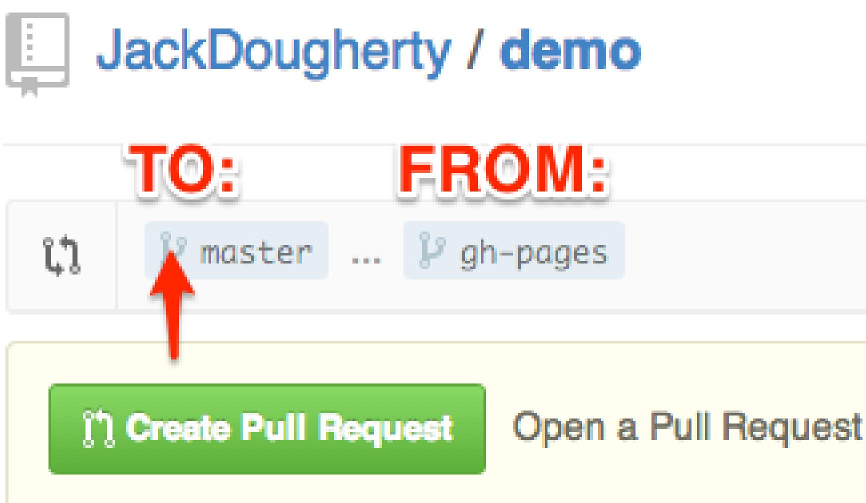
If a code template already exists somewhere on GitHub, "fork" a copy of the repository to your own account, create a gh-pages branch for the live web, and edit/modify the files as desired.

To sync and upload multiple files from your desktop to a GitHub repository, use the free GitHub for Mac or GitHub for Windows tool, which features a graphical user interface to do tasks above very easily. Advanced coders may use command-line instructions to upload and sync files to GitHub most efficiently.

12) To BACKUP and SHARE your work: After adding or editing files in the gh-pages branch, create a "pull request" (to merge files) to the master branch.



13) Use the drop-down menus to CAREFULLY select the direction of the merge in this TO-FROM format: TO master branch FROM gh-pages branch



14) Follow these steps to finish the pull request, which merges all content TO the master branch FROM gh-pages branch:

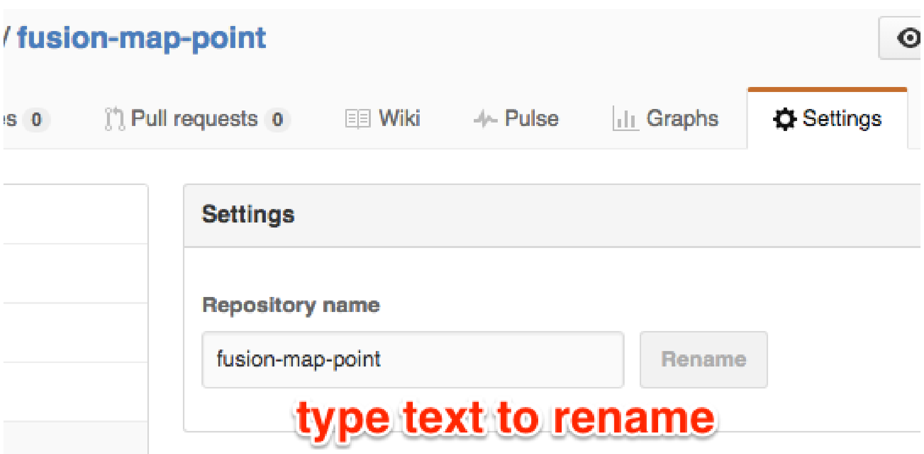
- Create the pull request (select green button)
- Give it a title (such as "updated index and ReadMe")
- Send pull request (green button)

- Merge pull request (another green button)
- Confirm merge (and another green button!)
- Always double-check your repository branch names to do editing and live web testing in the gh-pages branch, then make sync a copy to public share in the master branch.

To embed an interactive data visualization from a GitHub Pages host site into a secondary website (such as an organization's WordPress.org site), see the Embed iframe with WordPress tutorial in this book.

See my sample repositories, with links to live demo sites, at https://github.com/?tab=repositories. Learn more about GitHub general features to share and collaborate on other users' public code repositories.

TO DO Reminder: GitHub allows users to create **one fork** of a repo to your account. To create a second copy, go to the repo of your first copy, click Settings, and rename it. But if you rename your repo, you also will need to change any links you created to its live version in the next section.



Work more efficiently with Atom editor and GitHub Desktop

TODO: REVISE outdated page: While you can do **nearly** everything in this book with GitHub in your browser, several steps will be faster and more efficient with two related free tools:

- download Atom Editor from GitHub (https://atom.io)
- download GitHub Desktop for Mac or Windows (https://desktop.github.com)

UPDATE OLD INSTRUCTIONS

Download the free GitHub Desktop tool to sync and additional GitHub repos on your local Mac or Windows computer. GitHub allows users to create one fork of the basic Searchable Map template repository. To create a second template, or to move and edit multiple files for more advanced versions, download the GitHub for Mac/Windows tool. 1. Download the free tool: GitHub for Mac or GitHub for Windows 1. In the Searchable Map Template in GitHub, click **Clone** and save to your hard drive 1. In your GitHub browser, create a new repository for your second template, and select options to create a README.MD and license (recommended: MIT). 1. Clone your second template repository to your hard drive 1. In your hard drive, copy and paste the files from the cloned Searchable Map Template to your cloned second template. Replace the existing README.MD and license files. 1. In your GitHub for Mac/Windows tool, **Commit and Sync** your second template to your GitHub online account. Title the commit before clicking the button. 1. Refresh your browser to view the synced files in your GitHub account. Start at the top of these directions to remove an old gh-pages branch, create a new gh-pages branch, and edit files.

Fix Common GitHub and Code Errors

What happens if you cannot view your published GitHub repository, or if your code breaks and no longer displays what it was designed to show? These are common problems, especially for newer students, because accidentally clicking the wrong box or mistakenly erasing a single character (such as a semicolon) can make your visualization seem to vanish, even though your work is usually still there. Breaking your code—and figuring out how to fix it—is a great way to learn, even if it requires trial and error.

Safely Delete your GitHub Repo and Start Over

If you need to delete your GitHub repo and start over, here's a simple way to safely save your work:

- Go to the top-level of your GitHub repository, similar to `https://github.com/USERNAME/REPOSITORY`
- Click the green "Clone or Download" button, and select Download Zip to receive a compressed folder of your repo contents on your computer.
- In your GitHub repo, click on Settings (upper-right area) and scroll down to Delete This Repository.
- To prevent accidental deletions, GitHub requires you to type in the REPOSITORY name.
- Now you can start over in one of these ways:
  - If you wish to Create a Simple Web Page with GitHub Pages, follow that tutorial again.

  - OR
  - Fork another copy of the original GitHub repository to your account. After you create your copy, if you wish to add selected files that you previously downloaded to your computer, follow directions to Upload Code with GitHub in the second half of this tutorial in this book

Problems with Creating a Simple Web Page with GitHub Pages

If you followed the Create a Simple Web Page with GitHub Pages tutorial, it should have created two web links (or URLs):

- your code repository, in this format: `https://github.com/USERNAME/REPOSITORY`
- your published web page, in this format: `https://USERNAME.github.io/REPOSITORY`

Be sure to insert your GitHub username, and your GitHub repository name, in the general formats above.

These URLs are NOT case-sensitive, which means that `https://github.com/USERNAME` and `https://gitub.com/username` point to the same location.

My simple GitHub web page does not appear

- Make sure that you are pointing to the correct URL for your published web page, in the format shown above.
- Be patient. During busy periods on GitHub, it may take up to 1 minute for new content to appear in your browser.
- Do a "hard refresh" to bypass any saved content in your browser cache.
  - Ctrl + F5 (most Windows-Linux browsers)
  - Command + Shift + R (Chrome or Firefox for Mac)
  - Shift + Reload button toolbar (Safari for Mac)
- Test the link to your published web page in a different browser. If you normally use Chrome, try Firefox.
- On rare occasions, the GitHub service or GitHub Pages feature may be down. Check https://status.github.com.

My simple GitHub web page does not display my iframe

- If you followed the Create a Simple Web Page with GitHub Pages tutorial and inserted an iframe in the README.md file, it will appear in your published web page, under these conditions:
  - Ideally, your README.md should be the ONLY file in this GitHub repository

  - Any other files in your repo (such as index.html, default.html, or index.md) will block the iframe HTML code in your README.md from being published on the web. If you accidentally selected a GitHub Pages Theme, you need to delete any extra files it created: click each file, select trash can to delete it, and commit changes.

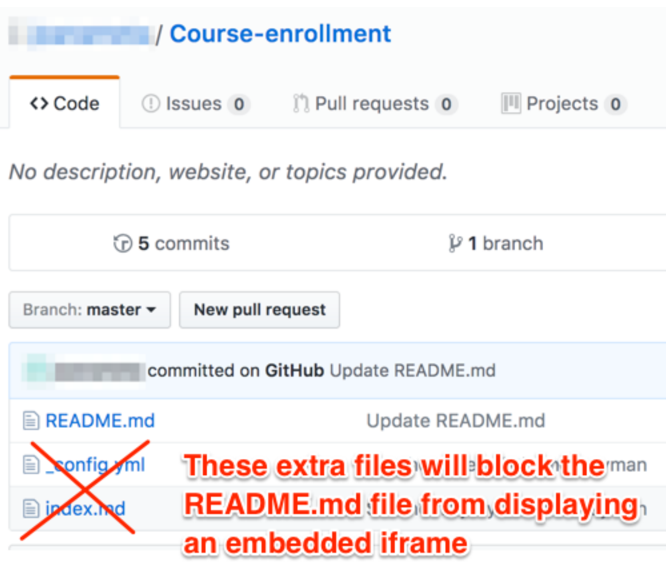


Figure 8.1: Screenshot: Extra files in GitHub repo will block iframe in your README

Problems with iframes

My iframe does not appear in my web page

- Go back to the Embed tutorials in this book to double-check the directions
- Items listed in your iframe (such as the URL, width, or height) should be enclosed inside straight quotation marks (single or double)
  - BROKEN iframe (missing quotation marks for width and height)

```
<iframe src="https://handsondataviz.github.io/leaflet-map-simple" width=90% height=350></ifr
```

  - FIXED iframe (with correct quotation marks)

```
<iframe src="https://handsondataviz.github.io/leaflet-map-simple" width="90%" height="350"><
```

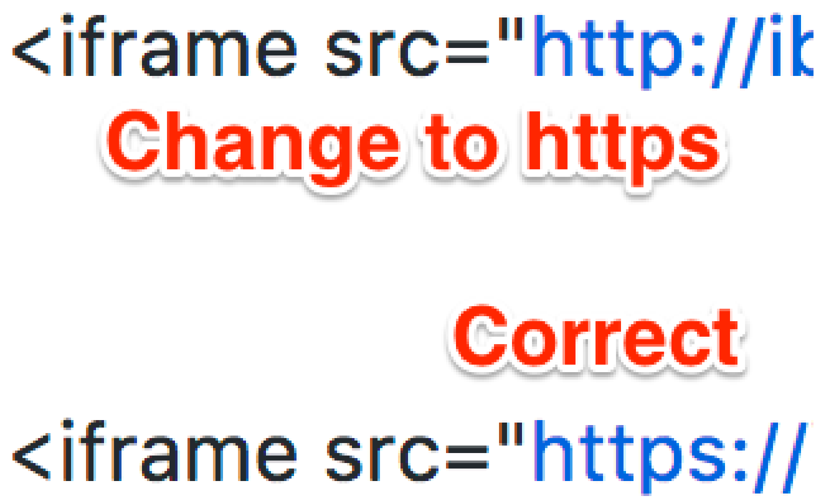


Figure 8.2: Screenshot: Replace http with https

- Use only `https` (the extra 's' means 'secure'), not `http`. Some web browsers will block content if it mixes http and https resources, and some code templates in this book require https.

- Use only straight quotes, not curly quotes. Avoid pasting text from a word-processor into GitHub, which can accidentally carry over curly quotes. Typing directly into the GitHub editor will create straight quotes.

Problems with Leaflet Maps with Google Sheets template

My map does not appear

1) Confirm that you have completed all of the key steps in the Leaflet Maps with Google Sheets tutorial in this book, especially these:

- Sign in to Google and File > Make a Copy of the Google Sheet to your Google Drive.
- File > Publish your Google Sheet (Jack often forgets this key step!)

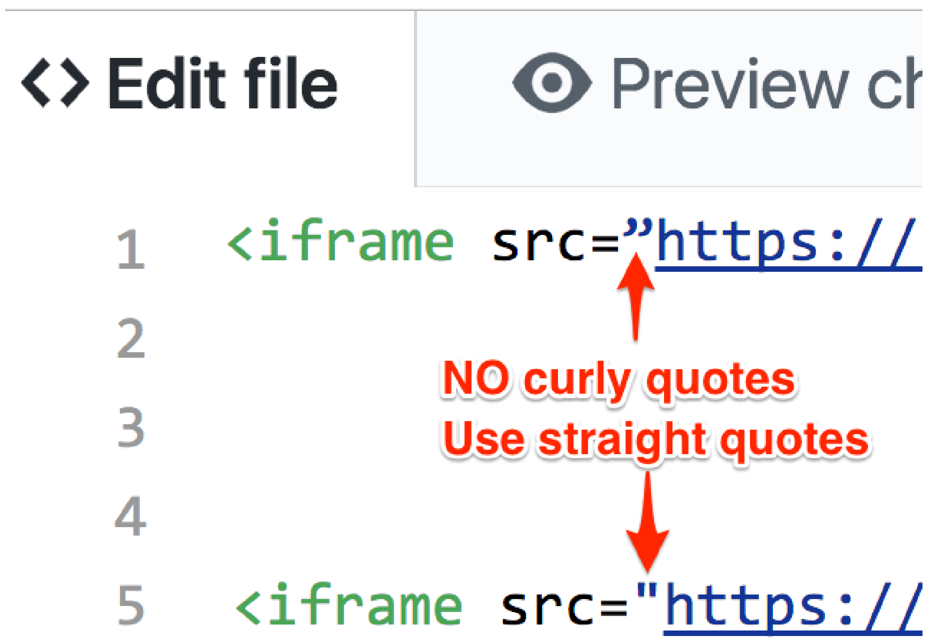


Figure 8.3: Screenshot: Curly quotes versus straight quotes

- Copy your Google Sheet web address from top of your browser (usually ends with `...XYZ/edit#gid=0`) and paste into your `google-doc-url.js` file in your GitHub repo. Do NOT copy the *Published* web address (which usually ends with `...XYZ/pubhtml`)
- When you paste your Google Sheet web address into `google-doc-url.js`, be careful not to erase single-quote marks or semicolon
- Go to your live map link, which should follow this format: `https://USERNAME.github.io/REPOSIT` refresh the browser, and wait at least 30 seconds.

2) Check your Google Sheet for errors:

- Do NOT rename column headers (in row 1) of any sheet, because the Leaflet Map code looks for these exact words.

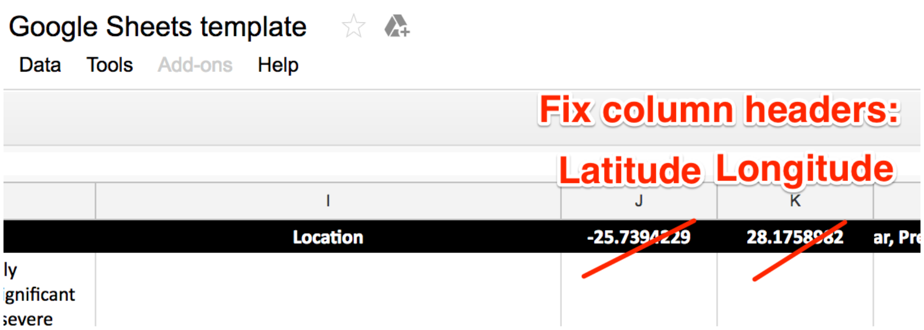


Figure 8.4: Screenshot: User accidentally renamed column headers in the Points tab

- Do NOT rename row labels (in column A) of any sheet, due to the same reason above.

- In your Points tab, DO NOT leave any blank rows

3) Confirm on GitHub Status (https://status.github.com/) that all systems are operational.

4) If you cannot find the problem, go to the top of this page to Safely Delete Your GitHub Repo and Start Over. Also, make a new copy of the Google Sheet template, give it a new name, and copy data from your old sheet using File > Paste Special > Values Only.

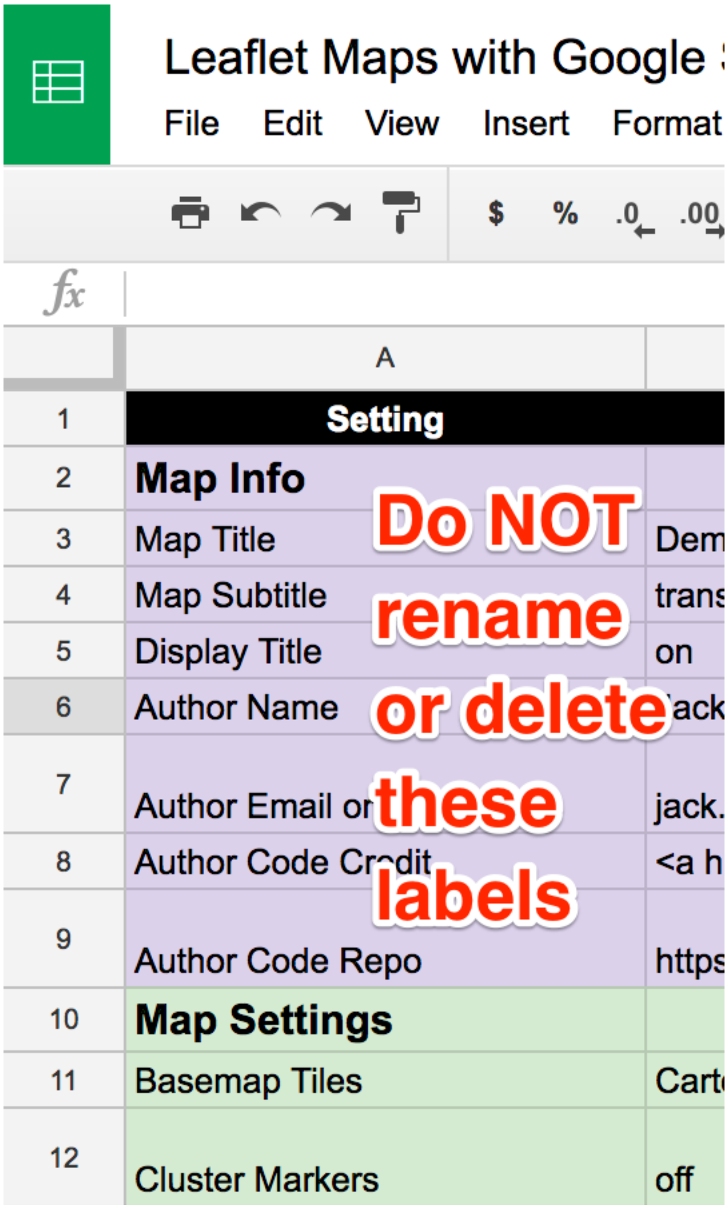


Figure 8.5: Screenshot: Do not rename or delete

Problems with Chart.js code templates

Chart displays old data If you upload new data to your Chart.js code template on GitHub Pages, and it does not appear in your browser after refreshing and waiting up to one minute, then GitHub Pages is probably not the cause of the problem. Instead, some browsers continue to show "old" Chart.js in the web cache. The simplest solution is to File > Quit your browser and re-open the link to your Chart.js

Problems with Mac Computers

No file extensions Several tools in this book will not work properly if your Mac Finder does not display file extensions. In other words, every file should include a period followed by several letters (such as data.csv and map.geojson). If you do not see these extensions at the end of each file name, then go to Finder > Preferences > Advanced and check the box to turn them on, as show below:

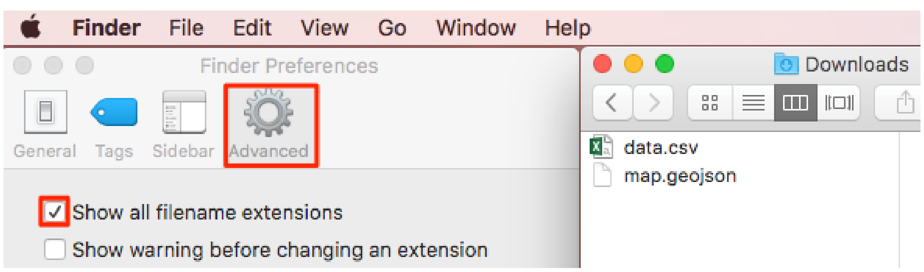
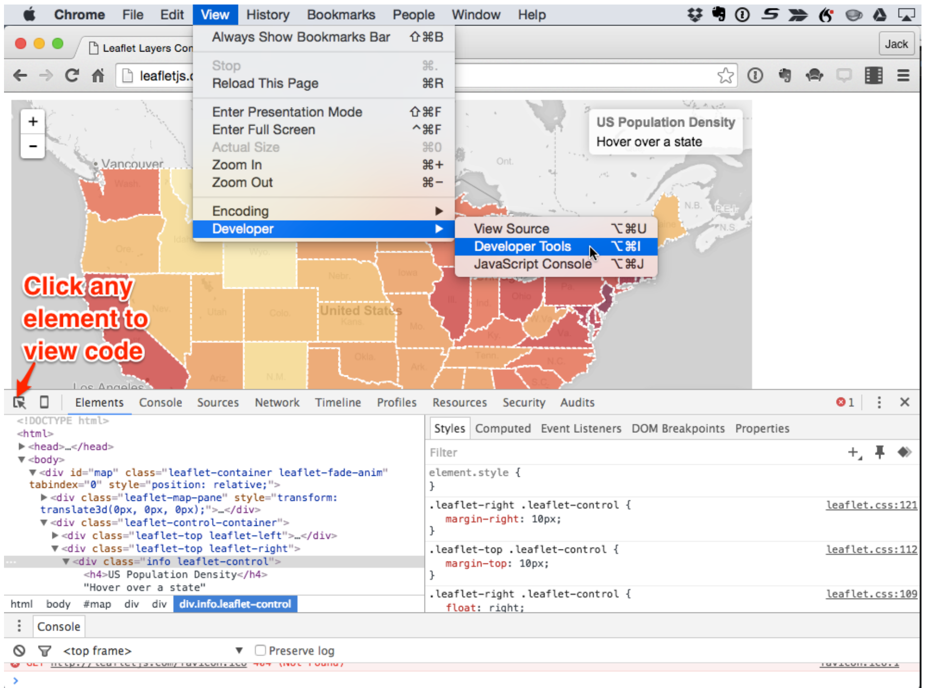


Figure 8.6: Screenshot: Checkbox to show filename extensions

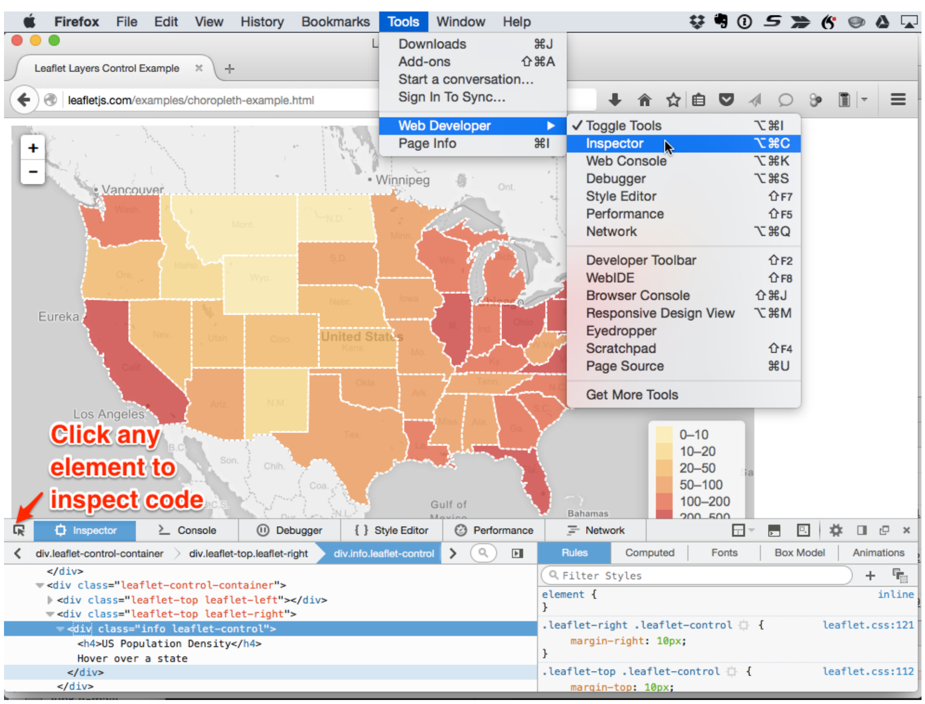
Solve Problems with Browser Developer Tools

Peek inside any website and view the web code under the hood with the browser developer tools.

In Chrome for Mac, go to View > Developer > Developer Tools



In Firefox for Mac, go to Tools > Web Developer > Inspector



Chapter 9

Transform Your Map Data

Interactive web maps are made up of different layers, such as background basemaps, colored or shaded polygons, and/or colored point markers. This chapter describes how to transform your data into layers that you can upload into online map tools and templates. Specifically, you will learn how to:

- Geocode locations into coordinates with US Census or Google
- Pivot address-level point data into polygon data
- Normalize data to create more meaningful polygon maps
- Convert map data with GeoJSON.io or Mapshaper.org
- Join spreadsheets and polygon boundaries with MapShaper.org

Enroll in our free online course **TO DO add link**, which introduces these topics in the brief video below, and offers more exercises and opportunities to interact with instructors and other learners.

Geocode Locations into Coordinates with US Census or Google

Many free map tools geocode locations by placing them on a map, such as Google My Maps tutorials in this book. But those tools typically do not allow you to easily extract the latitude-longitude coordinates for each point.

We created two free Google Sheets Geocoder scripts that have several advantages:

- convert locations (Hartford CT) or addresses (300 Summit St, Hartford CT) into latitude-longitude coordinates (41.748, -72.692) inside your Google Sheet

- show the location found in the geocoding database, and match quality, to review your results
- convert US addresses into US Census geography, such as census tracts, block groups, and blocks

As with any geocoding service, accuracy is not guaranteed. Inspect your results in the Found and Quality columns.

Google Sheets Geocoder: US Census or Google

- Geocode locations into latitude, longitude, with source and match quality, inside a Google Sheet
- Go to Google Sheet template, sign in to your account, and File > Make a Copy to your Google Drive https://docs.google.com/spreadsheets/d/1XvtkzuVyQ_7Ud47ypDJ4KOmz_5lOpC9sqeEDBbJ5Pbg/edit#gid=0
- Insert locations, select 6 columns, and select Geocoder menu: US Census or Google (limit 1000 daily per user)
- Google Sheets script will ask for permission to run the first time
- Note: The Leaflet Maps with Google Sheets template in this book includes this Geocoder script.

Figure 9.1: Screencast: Google Sheets Geocoder: US Census or Google

Google Sheets Geocoder: US Census Geographies

- Geocode US addresses into latitude, longitude, GeoID, census tract, inside a Google Sheet
- Go to Google Sheet template, sign in to your account, and File > Make a Copy to your Google Drive https://docs.google.com/spreadsheets/d/1x_E9KwZ88c_kZvhZ13IF7BNwYKTJFxbfDu77sU1vn5w/edit#gid=0

- Insert locations, select 8 columns, and select Geocoder menu: US Census 2010 Geographies
- Google Sheets script will ask for permission to run the first time

Figure 9.2: Screencast: Google Sheets Geocoder: US Census Geographies

About US Census 15-character GeoID

- Make sure that column G is formatted as text (to preserve leading zeros), not number
- Break down a sample GeoID: 090035245022001
  - state = 09
  - county = 003
  - tract = 524502 = 5245.02
  - block group = 2
  - block = 001

How it works The Google Sheet Geocoder runs from a script insert in the Google Sheet, which calls one of two free geocoding services:

- US Census Geocoder https://geocoding.geo.census.gov/geocoder. See more detailed documentation at http://www.census.gov/geo/maps-data/data/geocoder.html
- Geocode with Google Apps: The Maps Service of Google Apps allows users to geocode street addresses without using the Google Maps API, with a limit of 1,000 searches daily per user, https://developers.google.com/apps-script/reference/maps/geocoder

How to insert the Geocoder Script into any Google Sheet If you do not wish to File > Make a Copy of the Google Sheet templates above, you can insert the open-source Geocoder Scripts into your own Google Sheet:

- Go to Google Sheets Geocoder repo on GitHub
- Sign in to your Google Sheets, then select Tools > Script Editor
- File > Create New Script File
- Open and copy a script (such as geocoder-census-google.gs) and paste into your Script Editor
- Save and rename to geocoder-census-google.gs
- Refresh your Google Sheet and look for new Geocoder menu

TODO

- Also describe and link back how to split columns to form multi-columns addresses
- also describe and link back to how to unify columns to form a one-column address
- add this https://developers.google.com/maps/faq#geocoder_queryformat

See also: Batch upload to US Census

- Available at US Census Geocoder https://geocoding.geo.census.gov/geocoder/
- Upload CSV table with up to 1000 rows for faster processing, in this format, WITHOUT column headers:

AnyID | Street | City | State | Zip |
:—— | :—— | :— | :—- | : — |
1 | 300 Summit St | Hartford | CT | 06106 |

- Find Locations using > Address Batch (returns latitude, longitude coordinates)
- Find Geographies using > Address Batch (returns lat, lng, census geographies)
- Limitations:
  - Inputs and outputs have no column headers, which may confuse novices
  - Large batches may be delayed a few minutes during peak time periods
  - Unmatched addresses need to be manually corrected and re-submitted

Try it: Batch Upload to US Census

1) Right-click and Save this CSV file to your computer: sample-addresses-50. CSV means comma-separated values, a generic spreadsheet format that most data tools can easily open.

2) Use any spreadsheet tool to organize your address data into five columns: any ID number, street, city, state, zip code. **Remove all column headers**.

	A	B	C	D	E
1	1	300 Summit St	Hartford	CT	06106
2	2	960 Main St	Hartford	CT	
3	3	133 Allen Pl, Apt 2	Hartford	CT	06106

Hints:

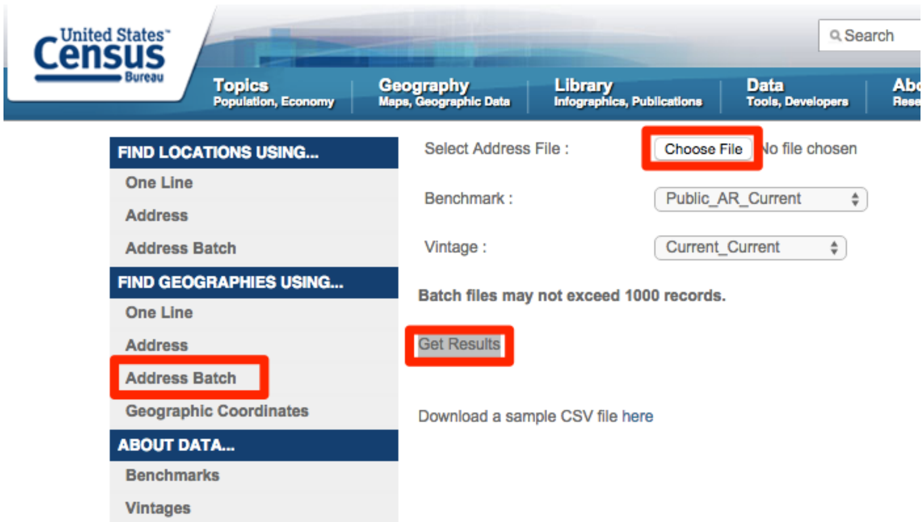
- If your data lacks ID numbers, quickly create a column of consecutive numbers, as shown in this book.
- If your address data includes apartment numbers, leave them in.
- Only the ID and address fields are required. City, state, and zip code may be blank if you lack any of this information, but fewer matches will be exact.
- If your address data is combined into one cell, such as: 300 Summit St, Hartford, CT 06106
  - then try to clean your data with the split column method in this book.
- If you need to temporarily move other non-address data columns into a second spreadsheet, remember to paste the column of ID numbers into the second sheet. After geocoding, sort both sheets by the ID column, then paste to rematch the data.

3) Save the file in CSV generic spreadsheet format, in batches of no more than 1,0000 rows per file. Learn more about saving in CSV format in this book.

4) Go to US Census Geocoder (https://www.census.gov/geo/maps-data/data/geocoder.html)

5) Select the Find Geographies Using… Address Batch button for maximum results, including lat-long coordinates and census geography (tracts and block groups). *If census geography is not needed, select Find Locations Using… Address Batch.*

6) Click the Choose button to upload your CSV file. Use the default benchmark and vintage settings for the most current data. Click the Get Results button, and be patient if using the service during busy weekday hours.



7) Census Geocoder will download the results through your web browser in a file named: GeocodeResults.csv. Since these results do not contain column headers, use the screenshot below for guidance, or read the Census Geocoder documentation for more details.

	A	B	C	D	E	F	G	H	I	J	K	L
1–2	ID	Input address	Quality		Match address	Lon,Lat	Tiger	St	Cty		Tract	BlkGp
3	338	21 Allen Place, Hartford, CT, 06106	Match	Exact	21 ALLEN PL, HARTFORD, CT, 06106	-72.6841,41.752377	3495001	L	9	3	502700	3003
4	339	22 Elliott Street Apt # 204, Hartford, CT, 06114	Match	Exact	22 ELLIOTT ST, HARTFORD, CT, 06114	-72.6743,41.74627	3516079	R	9	3	500100	2001
5	941	60 Ellsworth Street, 2nd Floor, Hartford, CT, 06114	Match	Exact	60 ELLSWORTH ST, HARTFORD, CT, 06114	-72.684685,41.745457	3515876	R	9	3	502700	2002
6	942	37 Shultas Place, Hartford, CT, 06114	Match	Exact	37 SHULTAS PL, HARTFORD, CT, 06114	-72.6754,41.748585	3516051	L	9	3	500100	2000
7	335	251 Lawrence Street, Hartford, CT, 06115	Match	Non_Exact	251 LAWRENCE ST, HARTFORD, CT, 06106	-72.687904,41.76278	3494808	L	9	3	503000	2000
8	330	19 Amity Street - Apt #1, Hartford, CT, 06106	No_Match									
9	331	280 Collins St., Apt. 202, Hartford, CT, 06105	Match	Exact	280 COLLINS ST, HARTFORD, CT, 06105	-72.694916,41.773575	3494393	R	9	3	524600	1005

8) Use a spreadsheet tool to open the CSV file. Sort results by the match quality (columns C and D), with these entries: match exact, match non-exact, tie, no-match.

9) For results without an exact match, check the address for typos, and try to re-geocode in a separate CSV file. The US Census Geocoder tool is very good, but not perfect. For a few rows of hard-to-match data, use a different geocoding tool, such as the Google Maps > What's Here feature described at the top of this page, to look up individual addresses and coordinates.

Learn more

- Aggregate individual rows of data into groups by census area with pivot tables.
- Download census data by tract or block group, and use the VLOOKUP formula to join or merge this rows of data that you have geocoded by census tract or block group.

Pivot Address-Level Point Data into Polygon Data

Problem: If I begin with address-level point data, how can I transform this into polygon map data?

One solution: In any spreadsheet, split your address data into separate columns (such as Street, City, State), then create a pivot table to aggregate rows into groups (such as the number of addresses in each City or State).

If your location data is combined into one column (example: 300 Summit St, Hartford CT), then see the Spreadsheets: Split Data Columns tutorial in this book.

Here's an example using a long list of US hospitals from the Medicare open data repository, which is already split into separate columns: https://data.medicare.gov/Hospital-Compare/Hospital-General-Information/xubh-q36u

1) Open the link above, see columns of data (Address, City, State, etc.), and click the blue Export button to download in the CSV generic spreadsheet format.

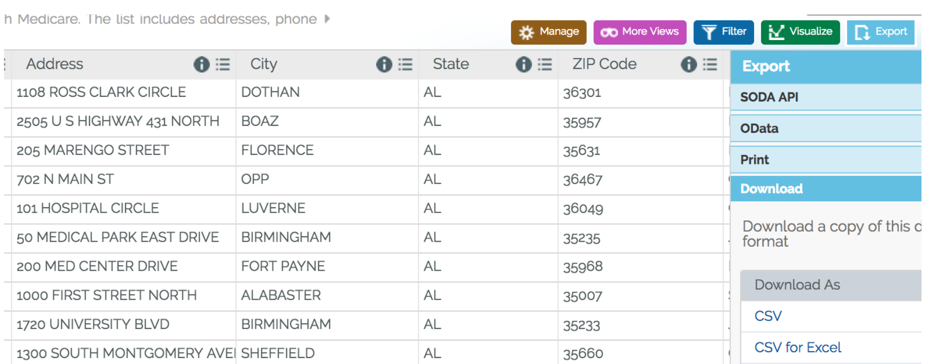


Figure 9.3: Screenshot: Export US Hospital data into CSV format

2) Open the file with any spreadsheet tool, and create a pivot table to count up the number of hospitals in each state. For help, see the Pivot Table tutorial in this book.

3) Now you can copy and paste the pivot table raw data of hospitals by US states. See the Normalize Data tutorial and also the Edit and Join Spreadsheet with Polygon Map using Mapshaper tutorial in this book.

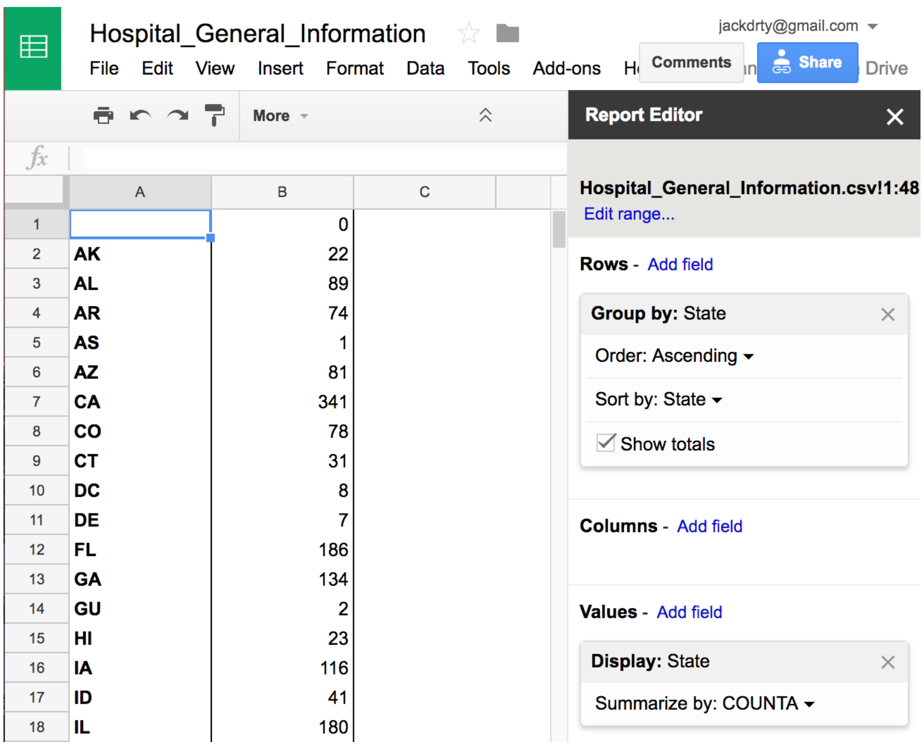


Figure 9.4: Screenshot: Pivot Table of US Hospitals by State

Other Solutions

- use the Google Sheets Geocoder: US Census Geographies tutorial in this book to convert addresses into census tracts, etc., and then pivot
- do a polygons-to-points spatial join with Mapshaper.org ** TO DO **

Normalize Data to Create Meaningful Polygon Maps

When preparing polygon maps, normalize your data to create more meaningful comparisons. Learn the difference between:

- **Raw data:** absolute values, such as the population of each US state (example: Connecticut population in 2015 = 3,590,886 people)
- **Normalized data:** represented on a standard scale (also known as standardized data), such as the population density of each US state (example: Connecticut 2015 population density = 3,590,886 people / 4,482 square miles = 742 people per square mile, equivalent to 1,922 people per square kilometer)

The difference between raw versus normalized data matters, especially in polygon maps. For example, the US states of Connecticut and Iowa have similar populations of about 3 million people each. But the rural midwestern state of Iowa has a much larger land area of over 55,000 square miles, while the more urbanized eastern state of Connecticut has a smaller land area of around 4,000 square miles. We can display all of this data in a table (as show below), but when making a polygon map, it makes most sense to show a normalized value, such as population density.

US State	Population 2015	Land Area (in square miles)	Density (pop per square mile)
Iowa	3,123,899	55,857	56
Connecticut	3,590,886	4,842	741

But raw data still matters, too. Although normalized data allows for easier comparisons across regions of different size, it can hide very low raw data values. For example, imagine two city neighborhoods with equally high unemployment rates of 20%, a normalized value. But if one neighborhood has a labor market population of 5,000 people while the other has only 500, the actual number of unemployed people in the second neighborhood is much smaller, as shown in the table below.

Neighborhood	Labor Market Population	Unemployment Rate	Actual Unemployed People
First	5,000	20%	1,000
Second	500	20%	100

Different ways to normalize data

After you understand the basic concept, also think about different ways to normalize the same data. Your method depends on the type of data story you wish to emphasize. Look at the table excerpt below on US population and land area by state in 2015:

	A	B	C
1	name	pop2015	land-area-sq-mi
2	Alabama	4858979	50645
3	Alaska	738432	570641
4	Arizona	6828065	113594
[redacted]			
52	Wyoming	586107	97093
53	Puerto Rico	3474182	3424
54	TOTAL	324893003	3535329

Figure 9.5: Screenshot: US population and land area

There are at least two acceptable ways to normalize this raw data:

- Normalized by area: Population per square mile in each state (calculate = pop / square miles)
- Normalized by total: Percent of total US population in each state (calculate = state pop / total US pop)

For example:

US State	Population 2015	Land Area (sq. mi)	Density (per square mile)
Connecticut	3,590,886	4,842	741

Convert to GeoJSON format

When you find map data, it may be stored in one of these common data formats below:

GeoJSON

GeoJSON is newer, popular open format for map data, and works across many tools, so is our top recommendation in this book. GeoJSON files can be used with Leaflet map code, Google Maps JS API code, Carto map tools, and more. Also, your GitHub repository will automatically display any GeoJSON files in a map view.

GeoJSON data must follow a structured format, but the file name may end with either `.geojson` or `.json`. The GeoJSON structured format orders coordinates in *longitude-latitude* format, the same as X-Y coordinates in mathematics. This is the opposite of Google Maps and several other web map tools, which order points in *latitude-longitude* format. For example, Hartford Connecticut is located at (-72.67, 41.76) in GeoJSON, but (41.76, -72.67) in Google Maps.

Shapefiles

The shapefile format was created in the 1990s by ESRI, the company that developed ArcGIS software. Shapefiles typically appear as a folder of subfiles with suffixes such as `.shp`, `.shx`, `.dbf`, and others. Although government agencies commonly distribute map data in shapefile format, the standard tools for editing these files—ArcGIS and its free and open-source cousin, QGIS—are not as easy to learn as other tools in this book. For this reason, this book recommends converting shapefiles into one of the more friendlier formats below.

Keyhole Markup Language (or KML)

The KML format rose in popularity during the late 2000s. Google Earth, a free and user-friendly tool, allowed many people to view and edit geographic data. KML files are commonly used in the Google Fusion Tables maps described in this book. Sometimes `.kml` files are distributed in a compressed `.kmz` format. See the chapter on converting from KMZ to KML format in this book.

GeoJson.io to Convert, Edit, and Create Map Data

TODO:

- rewrite into tool review and tutorial format
- place polygon conversion at top and specify import-export formats

Go to http://geojson.io to explore this open-source web tool to convert, edit, and create GeoJSON map data. The tool was originally developed by Tom MacWright, and is supported by Mapbox.com.

Convert a CSV spreadsheet of point data into GeoJSON

Use any spreadsheet tool and prepare a list of coordinate points (known as features). You must include column headers **lat** and **lon**, or a fuller spelling, such as *latitude* and *longitude*. The order of the columns does not matter. Also, you can add more headers to identify each point (example: name) and include more details (known as the properties of the features).

name	lat	lon	optional	optional2
Hartford	41.76	-72.67	city	https://en.wikipedia.org/wiki/Hartford,_Connecticut
Bloomfield	41.85	-72.73	suburb	https://en.wikipedia.org/wiki/Bloomfield,_Connecticut
West Hartford	41.76	-72.75	suburb	https://en.wikipedia.org/wiki/West_Hartford,_Connecticut

Save your spreadsheet in generic CSV format. *Hint:* see Save Spreadsheet as CSV chapter in this book.

Try it! Click this link and Save to download this sample file to your computer: name-lat-lon-info in CSV format. CSV means comma-separated values, a generic spreadsheet format that most data tools can easily open.

Drag the CSV file into the GeoJSON.io map window. Flip between the JSON and Table tabs to view or edit the data.

Figure 9.6: Screencast: GeoJson.io

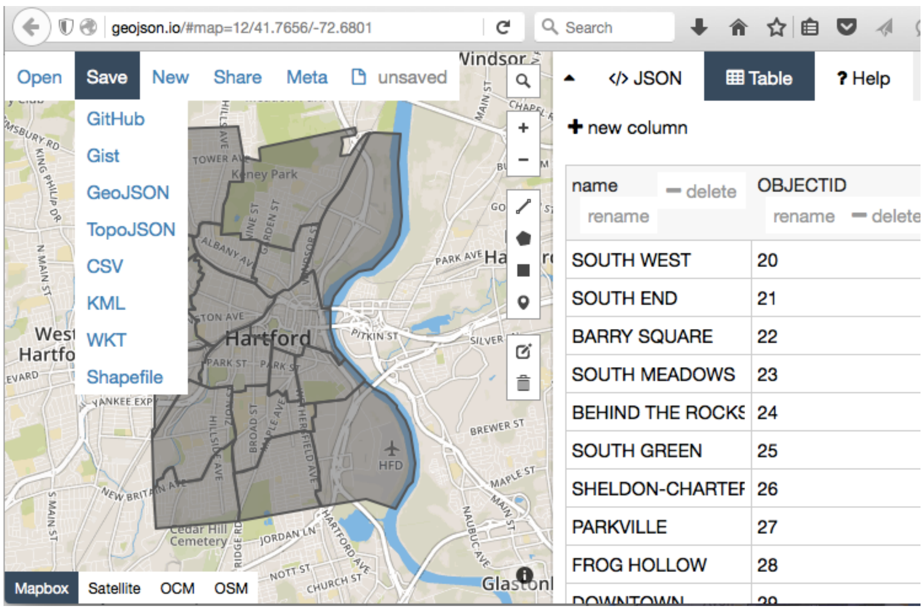
Select the Save menu and export into GeoJSON format.

Optional: Login to GeoJSON.io with your GitHub account and save directly to your repository.

Convert Shapefile or KML polygons into GeoJSON

Polygon boundary data is often shared as ArcGIS Shapefiles (.shp) or Keyhole Markup Language (.kml) files. Drag any of these (and other) files into the http://GeoJSON.io map window. Flip between the JSON and Table tabs to view or edit the data.

Select the Save menu and export into GeoJSON format.



Create GeoJSON data with drawing tools

Use the http://GeoJSON.io drawing tools to create points, polygons, and polylines. Flip between the JSON and Table tabs to view or edit the data.

Learn more about GeoJSON.io

Read about more advanced features and view the code at https://github.com/mapbox/geojson.io

MapShaper.org to Convert, Edit, and Join Data

TODO:

- rewrite into tool review/tutorial format

- put conversion at top and clarify import-export formats
- recommended browsers: Firefox or Chrome
- Mac users: go to Finder > Preferences > Advanced > turn on Show file extensions

MapShaper (http://MapShaper.org) is another versatile open-source mapping tool, developed and maintained by Matthew Bloch on GitHub. Using the web interface, users can:

- Import and export map layers in multiple formats: Shapefile, GeoJSON, CSV, and more
- Simplify (or smooth out) geographic details to reduce map file size
- Edit geography with powerful commands (dissolve, clip, join files, etc.)

This free and easy-to-learn MapShaper.org web tool has replaced *many* of my map preparation tasks that previously required expensive and hard-to-learn ArcGIS software, or its free but still-challenging-to-learn cousin, QGIS. Even advanced GIS users may discover MapShaper.org to be a quick alternative for some common time-consuming tasks.

The examples below focus on polygon boundary data to illustrate common map editing tasks. But MapShaper.org also works with other data layers, such as tables, points, and lines.

Import and convert map boundary files

Try it! Right-click the link and Save to download this sample file to your computer: ct-towns in GeoJSON format. If you accidentally open a page of GeoJSON code in your browser, select File > Save Page As to download to your computer.

1. Drag-and-drop any map layer into the http://MapShaper.org browser window.

- Import GeoJSON (.geojson or .json), TopoJSON, CSV, or Shapefile formats
- For Shapefiles, import the .shp (features), .dbf (attribute data), and .prj (projection) files. Reminder: the WGS84 projection is most portable across multiple platforms.
- KML/KMZ files are not compatible. To convert these into a format that Mapshaper can import, see the Convert KMZ to KML and Geojson.io chapters in this book.

2. Click the Export button and select your preferred format:

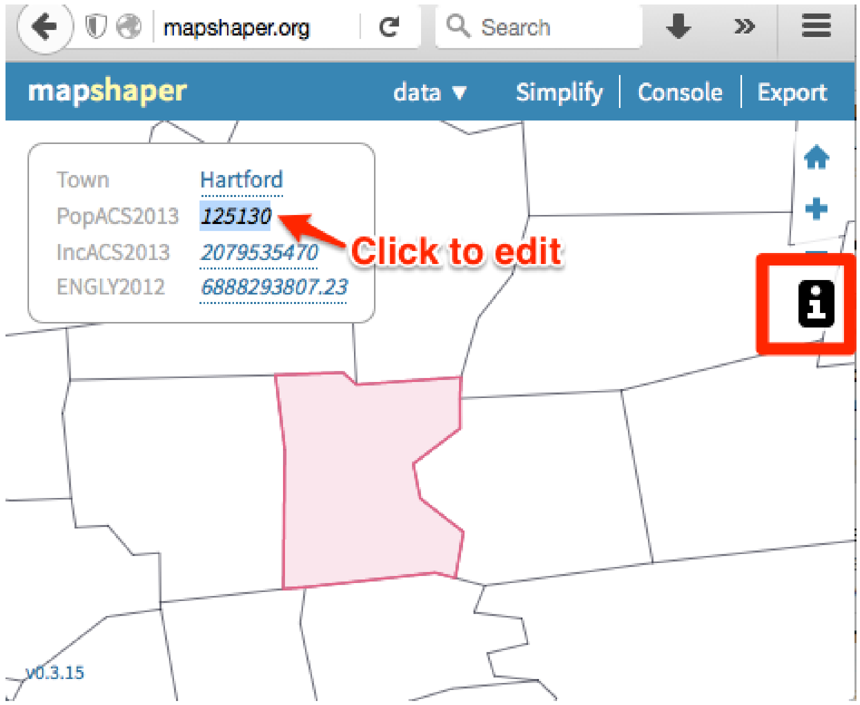
- Shapefile (best for ArcGIS/QGIS software)
- GeoJSON (best for Leaflet and GitHub tools in this book)
- TopoJSON (similar to GeoJSON, with topographical data)
- SVG (Scalable Vector Graphics, for print or online)
- CSV (Comma Separated Values, generic spreadsheet format)

Figure 9.7: Screencast: Mapshaper convert

Edit data for specific polygons

To edit data for any polygon in MapShaper.org:

- Click the “i” information button
- Select the polygon
- Click inside its pop-up info window to directly edit the data

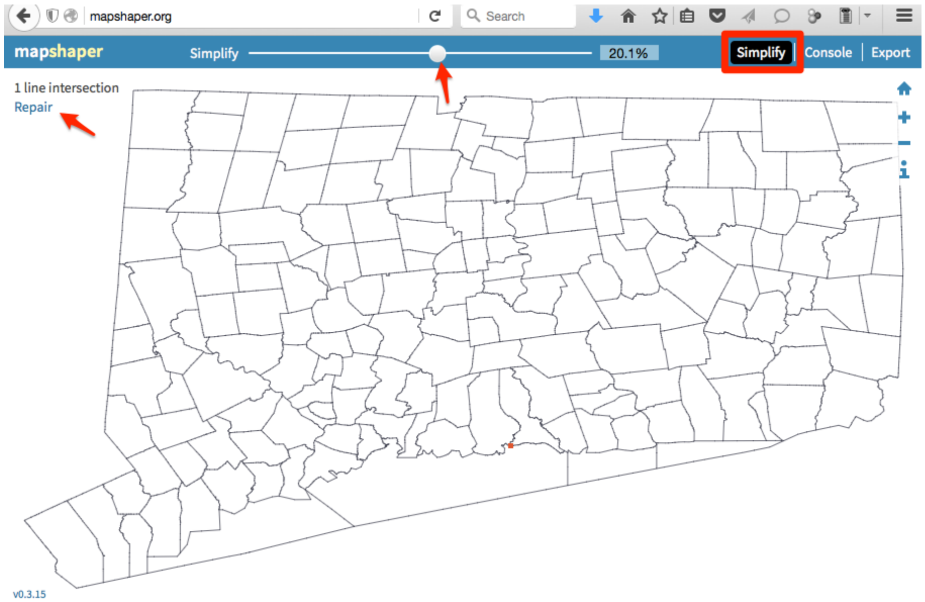


Simplify map boundaries to reduce file size

If your data visualization project displays a zoomed-out state or national or world map, small geographic details that are invisible at these zoom levels may not be necessary. Consider using the Simplify command to reduce the file size, which may help your interactive web map to load faster for web visitors. The example below began with a detailed map of Connecticut town boundaries (1:100,000 scale) at 2MB, which MapShaper simplified – without visibly sacrificing details at the statewide zoom level – to a reduced size of about 200KB.

1. Try it! Download and upload the sample GeoJSON file as described in the Import section above.
2. Click the Simplify button to review options, and in most cases, accept the default settings. Click Next.
3. Slide the Simplify button from 100 percent down to an appropriate number for your map zoom level. If important geographic details disappear, you may have gone too far.
4. Look in the upper-left corner and click on recommended Repairs to your map file.

5. Complete the process by clicking Simplify once again. Export your file in the preferred format for your project.



Dissolve internal polygons to create an outline map

MapShaper.org also includes a Console button to type in commands for common map editing tasks. Imagine that you begin with a boundary map that includes internal polygons, but your goal is to remove all of them to create an outline map.

Click the Console button, which opens a window to type in commands. Enter the command below, then press return. Close the Console window and Export your outline map.

```
-dissolve
```

Clip a map to match an outline layer

Imagine that you start with a polygon map of all towns in Connecticut, and an outline map of Hartford County, a larger region that includes some (but not all) of those smaller towns. Your goal is to create a polygon map of all towns inside Hartford County. In other words, we will "clip" the statewide town map using the county outline map.

Figure 9.8: Screencast: Mapshaper dissolve

Try it! Right-click the link and Save to download both sample files to your computer:

- ct-towns in GeoJSON format
- hartfordcounty-outline in GeoJSON format
- If you accidentally open a page of GeoJSON code in your browser, select File > Save Page As to download to your computer.

Refresh the browser to start a new session in http://MapShaper.org.

1. Drag-and-drop the ct-towns.geojson file to import to MapShaper.

2. Drag-and-drop the hartfordcounty-outline.geojson map to MapShaper, and click Import to add this second layer.

3. In the drop-down menu, select the first map (ct-towns) to display it as the active layer.

4. Click the Console button, type or paste in the command below, and press enter.

```
-clip hartfordcounty-outline.geojson
```

5. The command above instructs MapShaper to clip the active map layer (ct-towns) using the second layer (hartfordcounty-outline).

6. Sometimes the boundaries of the clip layer do not precisely match up with your active layer, due to differences between their sources. If necessary, add the `cleanup` command to remove any null features or small "slivers" that remain after the clip.

```
- clip hartfordcounty-outline.geojson cleanup
Removed 3 null features and 5 slivers
```

TO DO fix animation to match new file names

Figure 9.9: Screencast: Mapshaper clip

Remove unwanted data columns

Sometimes your polygon map contains several columns of unwanted data. To quickly remove them, enter the "-filter-fields" Console command to keep only the columns you list. The example below deletes all columns *except* "town":

```
-filter-fields town
```

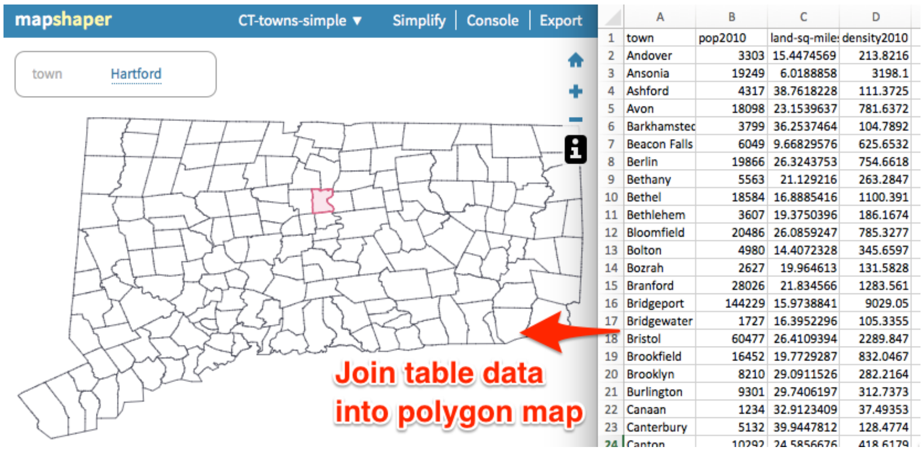
Join spreadsheet data with polygon map

** TO DO ** - fix images and animations to map the new file names and column headers

A common mapping task is to join (or merge) new data columns into a polygon boundary map, and MapShaper.org makes this very easy. Imagine that you have two files:

- Connecticut town boundary map
- a spreadsheet of town population data

Your goal is to unite these files, so that you can later display them in a thematic polygon map. Since these two files share a common column of data – the town names – you can join them together into one merged file.



Try it! Right-click each link and Save to download two sample files to your computer:

- ct-towns in GeoJSON format
- ct-towns-popdensity in CSV format

If you accidentally open a page of GeoJSON code in your browser, select File > Save Page As to download to your computer.

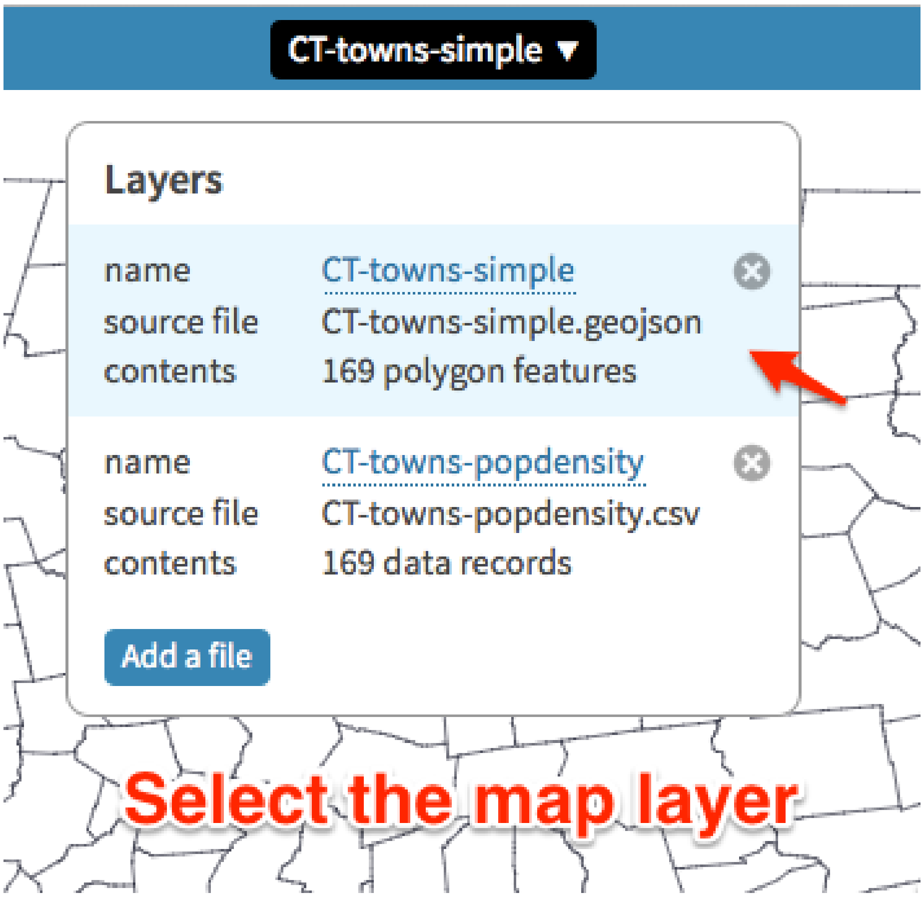
Refresh the browser to start a new session in http://MapShaper.org.

1. Drag-and-drop the ct-towns.geojson boundary file into MapShaper. Select the "i" info button and click on any polygon to confirm that the column header is "name".

2. Open the ct-towns-popdensity.csv file with any spreadsheet tool and confirm that first column header also is "name". Close this file.

3. Drag-and-drop the ct-towns-popdensity.csv file into MapShaper.org, and select the Import button to add it as a second layer. This table layer will appear as rectangular cells, because it does not contain geographic information.

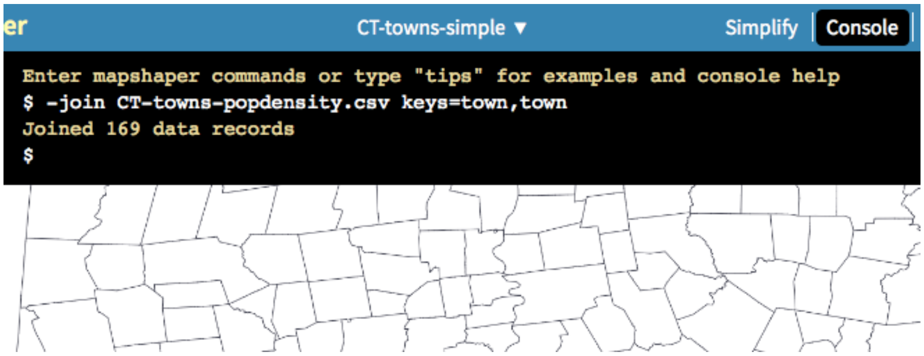
4. Click the drop-down menu and select the map to display it as the active layer.



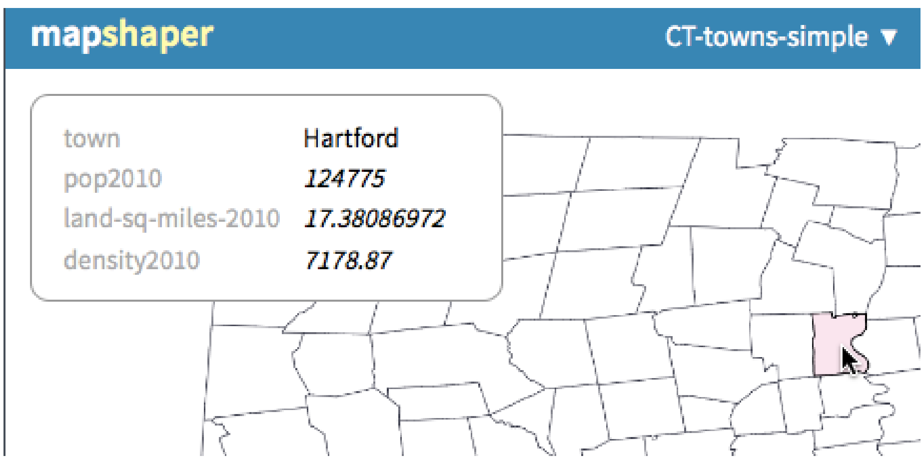
5. Click the Console button, type this command, and press return:

```
-join ct-towns-popdensity.csv keys=name,name
```

Type this precisely, with **no spaces** between the words in your keys. This command instructs MapShaper to join the active map layer to the CSV table layer, based on their shared column of data, labeled as "name" in both files. In this example, 169 rows are merged together.



6. Click the Console button to close the command window. Select the "i" info button and click any polygon to confirm that it now contains the new table data. Export the file in your preferred format.



More about joins

1. If you don't have a CSV table that matches the columns in your boundary map data, you can easily create one. Upload the boundary map to MapShaper.org, and export in CSV format, and open with any spreadsheet tool. To match data columns in the CSV spreadsheet, use the VLOOKUP method in this book.

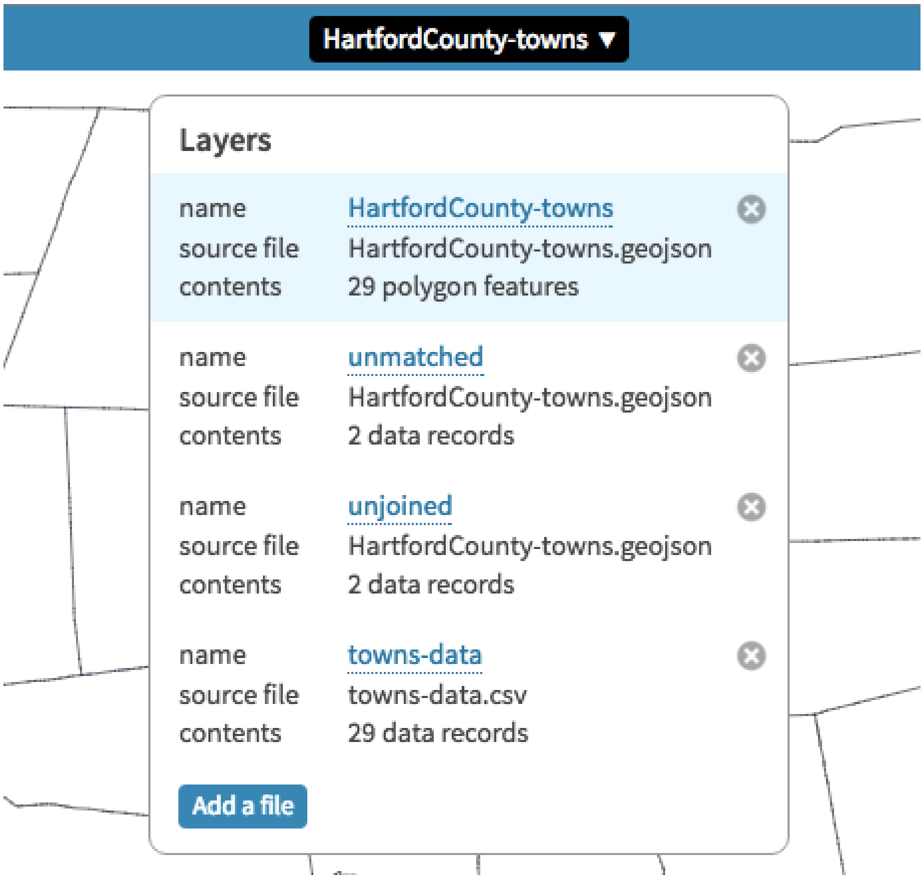
2. The simple join example above uses identical keys (name,name) because the two columns headers are the same. But if you need to join data where the headers are not the same, enter the first key (the polygon map) and the second key (the CSV table).

3. Mapshaper also helps you to keep track of data that are not properly joined or matched. For example, if the polygon map contains 169 rows (one for each town in Connecticut), but the CSV table contains only 168 rows of data, Mapshaper will join all of those with matching keys, and then display this message:

```
Joined 168 data records
1/169 target records received no data
```

To capture data records that are not properly joined, add these terms at the end of your join command: `unjoined unmatched -info`. The first term saves a copy of each unmatched record from the target table to a new layer named "unmatched," and the second term saves a copy of each unjoined record from the source table into another layer named "unjoined." In the example below, see the console command and results, and a screenshot of the two new layers.

```
$ -join towns-data.csv keys=name,name unmatched unjoined -info
Joined 27 data records
2/29 target records received no data
2/29 source records could not be joined
Layer 1 ...
```



Merge selected polygons with join and dissolve commands

** TO DO ** fix screenshots to match new data files and column headers

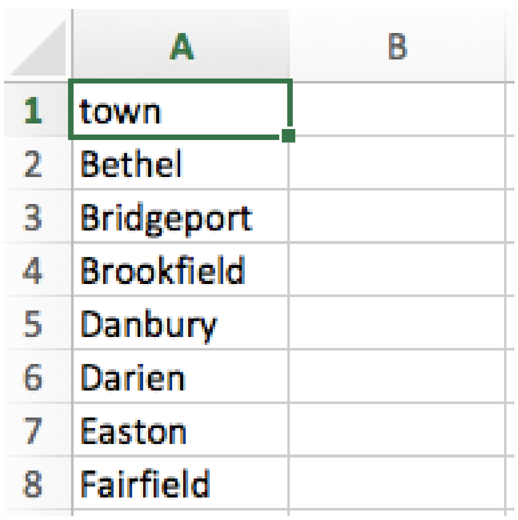
Another common task is to merge selected polygons in a boundary map, which you can do in MapShaper with the join and dissolve commands you learned above. Imagine that you wish to create regional "cluster" boundaries from smaller polygon areas. For example, the Connecticut Department of Public Health has grouped individual towns, such as Bloomfield and West Hartford, into regional health districts. Your task is to begin with a statewide polygon map of all town boundaries, and to create a new polygon map that displays these regional clusters.

Try it! Right-click the link and Save to download the sample files to your computer: ct-towns in GeoJSON format. If you accidentally open a page of GeoJSON code in your browser, select File > Save Page As to download to your computer.

Refresh the browser to start a new session in http://MapShaper.org.

1. Import the ct-towns.geojson map file into http://MapShaper.org.

2. Export in CSV format. This steps creates a spreadsheet that lists all of the polygon town names, without geographic details.



	A	B
1	town	
2	Bethel	
3	Bridgeport	
4	Brookfield	
5	Danbury	
6	Darien	
7	Easton	
8	Fairfield	

3. Open the CSV file with any spreadsheet tool. Copy the contents of the “name” column, paste it into a second column, and change the header of this second column to “merged”.

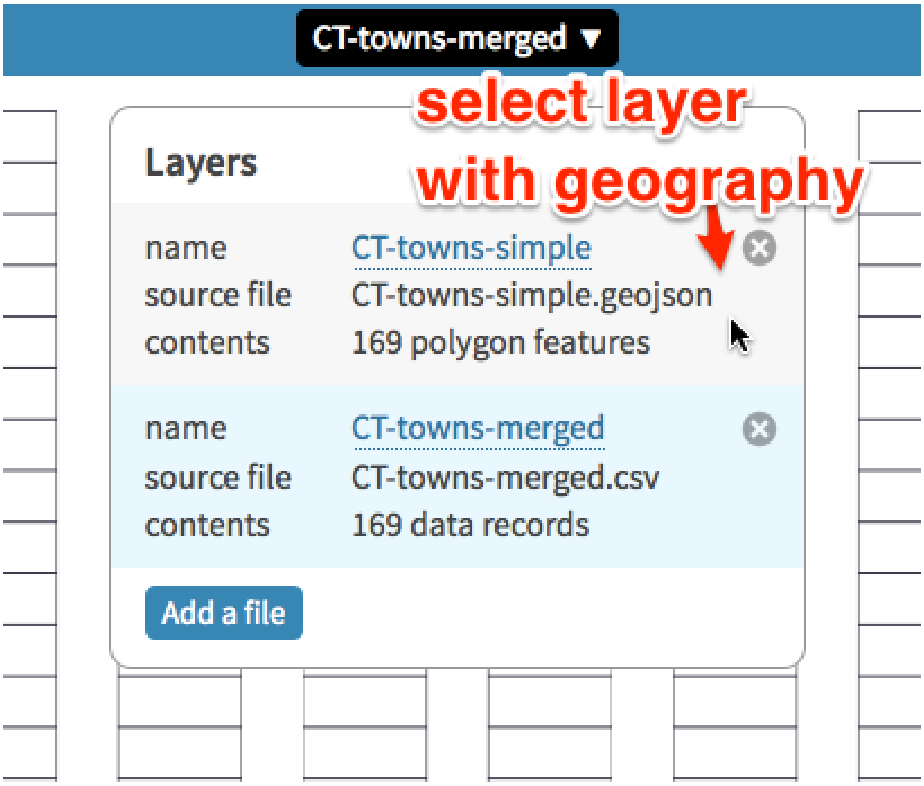
4. In the new “merged” column, create new listings for towns you wish to merge together. In this example, Bloomfield and West Hartford are merged into Bloomfield-West Hartford. Leave other towns unchanged.

	A	B
1	town	merged
2	Bloomfield	Bloomfield-West Hartford
3	West Hartford	Bloomfield-West Hartford
4	Bethel	Bethel
5	Bridgeport	Bridgeport
6	Brookfield	Brookfield
7	Danbury	Danbury
8	Darien	Darien
9	Easton	Easton

5. Save this spreadsheet in CSV format with a new file name, such as: ct-towns-merged.csv.

6. Drag this new ct-towns-merged.csv file into MapShaper, and click Import.

7. Use the drop-down menu to manage multiple layers in MapShaper. Since the CSV file has no geography, it appears as a series of rectangular cells. Instead, select the ct-towns.geojson map to display it as the active layer.

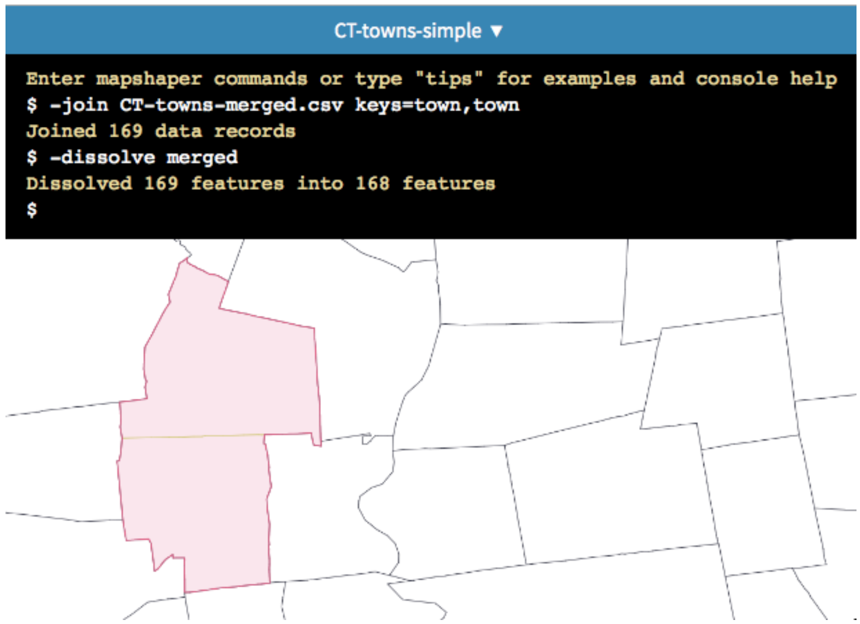


8. Click on the Console button, type in both of the commands below, and press Return at the end of each line:

```
-join CT-towns-merged.csv keys=name,name
-dissolve merged
```

How to understand the commands above:

- The first line "joins" the active layer (the polygon map) to the CSV spreadsheet, with "keys" to match their shared data columns, which are both labeled as "name".
- The second line dissolves the polygons of towns listed in the "merged" column of the CSV file. In this simple example, only Bloomfield and West Hartford are dissolved into a combined "Bloomfield-West Hartford" regional health district, and all of the other polygons remain the same.



Click the Console button to close its window. Select the "i" information button to inspect your merged polygons. Export the map in your preferred format.

Learn more advanced MapShaper methods

- See the MapShaper GitHub project wiki (https://github.com/mbloch/mapshaper/wiki/) for more command references and tips about map simplification

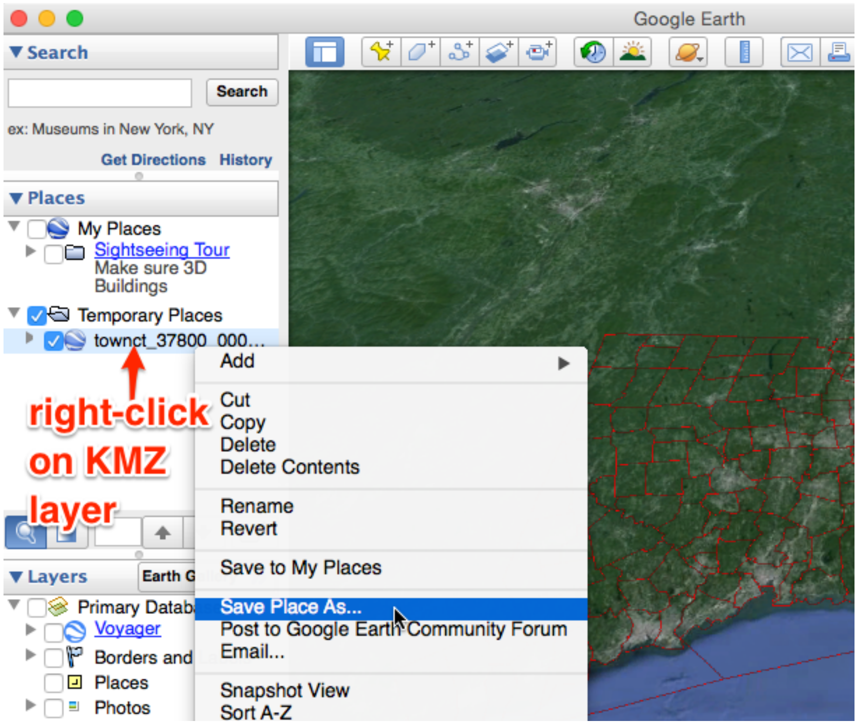
TO DO: illustrate concept of a point-to-polygon spatial join. When using the join command, "If the keys= option is missing, Mapshaper will perform a point-to-polygon or polygon-to-point spatial join."

Convert a Compressed KMZ file to KML format

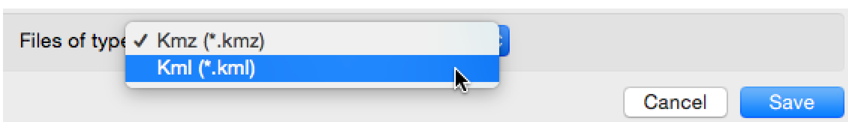
Sometimes KML files are distributed in compressed KMZ format. One easy conversion method:

- Install the free Google Earth tool
- Double-click on any .kmz file to open it in Google Earth

- Right-click (or control-click) on the .kmz layer and select *Save As*



- Use the *Save As* drop down menu to select .kml format



Or use any zip-utility and simply unzip the kmz file. Kmz is simply a 'zipped' version of a kml file!

Chapter 10

Detect Bias in Data Stories

While we like to believe data visualizations simply "tell the truth," when you dig further into this topic, you realize that there are multiple ways to represent reality. In this chapter, you will learn how visualizations display the biases of the people and the software that create them. Although we cannot stop bias, we can teach people to look for and detect it, and be aware of our own.

Sections in this chapter:

- How to Lie with Charts, inspired by Darrell Huff (1954)
- How to Lie with Maps, inspired by Mark Monmonier (1996)

Enroll in our free online course *TO DO add link*, which introduces these topics in the brief video below, and offers more exercises and opportunities to interact with instructors and other learners.

Learn more

- Darrell Huff, How to Lie with Statistics (W. W. Norton & Company, 1954), http://books.google.com/books?isbn=0393070875
- Mark S. Monmonier, How to Lie with Maps, 2nd ed. (University of Chicago Press, 1996), http://books.google.com/books?isbn=0226534219
- Nathan Yau, "How to Spot Visualization Lies," FlowingData, February 9, 2017, http://flowingdata.com/2017/02/09/how-to-spot-visualization-lies/

How to Lie with Charts

One of the best ways to learn how to detect bias in data visualization is to intentionally manipulate a chart, and tell two (or more) opposing stories with the same data. You'll learn what to watch out for when viewing other people's charts, and think more carefully about the ethical issues when you design your own.

This exercise was inspired by a classic book published more than fifty years ago: Darrell Huff, *How to Lie with Statistics* (W. W. Norton & Company, 1954), http://books.google.com/books?isbn=0393070875

Right-click this link and Save to download this sample data in CSV format to your computer: us-gross-domestic-product-per-capita. This historical data on economic productivity comes from the World Bank, World Development Indicators, http://data.worldbank.org/data-catalog/world-development-indicators

Upload the CSV file to your Google Drive (with Settings to Convert to Google format) to create a Google Sheet.

Select the data cells and Insert > Chart > Line chart, similar to the default version shown below:

In your Google Sheet chart, double-click the vertical y-axis to edit the Minimum and Maximum values.

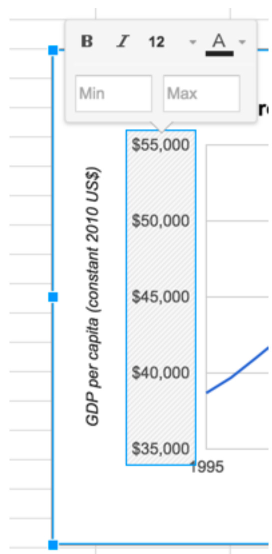


Figure 10.1: Screenshot: Edit the Min and Max values of the Y-axis

Make the line look "flatter" (slower economic growth) by lowering the minimum to $36,000, and increasing the maximum to $100,000, as shown below:

Make the line look like a "sharper increase" (faster economic growth) by increasing the minimum to $38,000, and lowering maximum to $52,000, as shown below:

** TO DO – add conclusion **

How to Lie with Maps

One of the best ways to learn how to detect bias in data visualization is to intentionally manipulate a map, and tell two (or more) opposing stories with the same data. You'll learn what to watch out for when viewing other people's maps, and think more carefully about the ethical issues when you design your own.

This exercise was inspired by Mark S. Monmonier, *How to Lie with Maps, 2nd ed.* (University of Chicago Press, 1996), http://books.google.com/books?isbn=0226534219

First, scroll through this data on Median Household Income for Hartford-area towns, 2011-15, from American Community Survey 5-year estimates. Or right-click to open this Google Sheet in a new tab.

Next, explore two different polygon maps of the same data. Use the drop-down menu to compare "Extreme Differences" versus "Uniform Equality"

Why are these two maps portray the same data so differently? To see the answer, look at the data ranges. . ..

** TO DO **

Create your own version…

Chapter 11

Tell Your Data Story

TODO: Write this chapter: Tell the story about your data, including its most meaningful insights and limitations Write compelling titles, labels, and sentences to accompany your visualization. Call attention to the most meaningful insights in your chart, and explain any data limitations.

This chapter draws inspiration from Cole Nussbaumer Knaflic, *Storytelling with Data: A Data Visualization Guide for Business Professionals* (Wiley, 2015), http://www.storytellingwithdata.com/book/

- Beginning, Middle, and End
- Draw Attention to Meaning

Chapter 12

Chart.js Code Templates

While beginners appreciate the drag-and-drop chart tools and tutorials described earlier in this book, such as Google Sheets and Tableau Public, more advanced users may wish to customize their visualizations, add more complex data, and control exactly how and where their work appears on the web. A more powerful and relatively easy-to-learn solution is to use code templates built with Chart.js https://www.chartjs.org/, an open-source library, which you can modify and host on GitHub, as described in this book.

Working with Chart.js

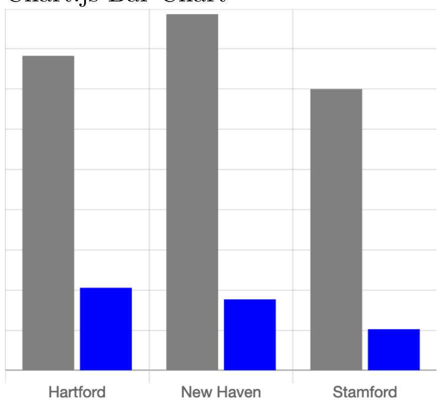
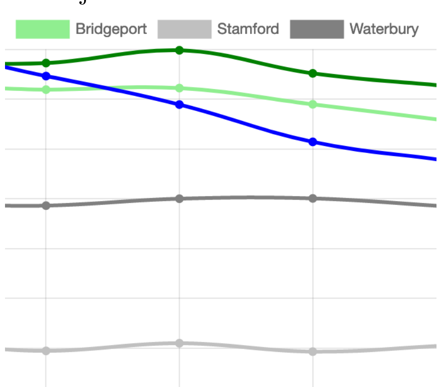
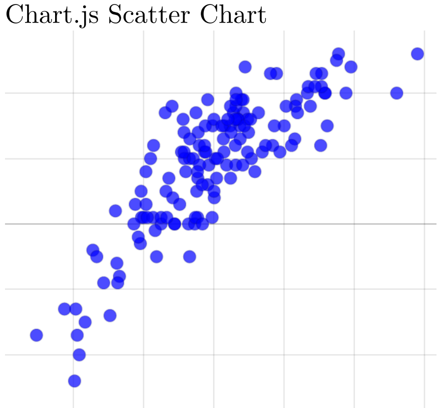
Pros

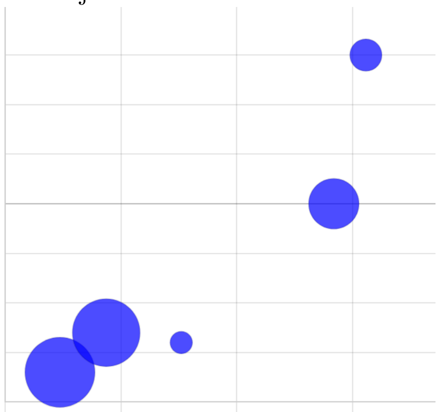
- Open-source code that is distributed under MIT license and is free for all and
- Easier for beginners to understand than more complex visualization code libraries such as D3.js

12.0.0.0.1 Cons

- Must host your own code repositories to publish to the web (with a free service such as GitHub Pages)

12.0.0.1 List of Chart.js templates

Templates	Best use and tutorial chapters
Chart.js Bar Chart 	Bar charts (vertical bar charts are often called column charts) can be used to compare categorical data. Template with tutorial: Bar Chart.js with CSV Data
Chart.js Line Chart 	Line charts are normally used to show trends (temporal data). Template with tutorial: Line Chart with CSV Data
Chart.js Scatter Chart 	Scatter charts (also scatterplots) are used to display data of 2 or more dimensions. Template with tutorial: Scatter Chart with CSV Data

Templates	Best use and tutorial chapters
Chart.js Bubble Chart 	Bubble charts are used to display data of 3 or more dimensions. Template with tutorial: Bubbble Chart with CSV Data

Inside the templates

The templates featured above vary from simple to complex, but all of them rely on four basic pillars:

- HTML: language to structure content on the web (example: index.html)
- CSS, or Cascading Style Sheet: to shape how content appears on the web (example: style.css)
- JavaScript: code to create the chart and interactivity (example: script.js)
- CSV: data that powers the visualization that is expressed in comma-separated format (example: data.csv)

Also, these templates refer to other code elements:

- library: link to online instructions to complete routine tasks (example: Chart.js)
- data: content to appear in chart, typically in CSV format (example: data.csv) or pulled from Google Sheets

Learn more

- Chart.js Samples, https://www.chartjs.org/samples/latest/

Bar Chart.js with CSV Data

Bar charts (vertical bar charts are often called *column charts*) can be used to compare categorical data. The y-axis (or x-axis for horizontal bar chart) should always start at 0.

Demo: https://handsondataviz.github.io/chartjs-templates/bar-chart/index.html

Source and instructions: https://github.com//chartjs-templates/tree/master/bar-chart

Line Chart.js with CSV Data

Line charts are often used to show temporal data (trends). The x-axis often represents time intervals. Unlike column or bar charts, y-axes of line charts do not necessarily start at 0.

Demo: https://handsondataviz.github.io/chartjs-templates/line-chart/index.html

Source and instructions: https://github.com//chartjs-templates/tree/master/line-chart

Scatter Chart.js with CSV Data

Scatter charts (also *scatterplots*) are used to display data of 2 or more dimensions. The scatter chart below shows the relationship between household income and test performance for school districts in Connecticut. Using x- and y-axes to show two dimensions, it is easy to see that test performance improves as household income goes up.

Demo: https://handsondataviz.github.io/chartjs-templates/scatter-chart/index.html

Source and instructions: https://github.com//chartjs-templates/tree/master/scatter-chart

Going beyond two dimensions

To show more than two dimensions in scatter charts, one can:

- **color** each data point differently to show third dimension, eg use shades of red and green to show 5-year trend in test performance;
- **resize** each data point to display fourth dimension, eg number of students in each school district;
- use different **icons or glyphs** to display fifth dimension, eg circles for male students and squares for female students.

Remember not to overwhelm the viewer and communicate only the data that are necessary to prove or illustrate your idea.

Bubble Chart.js with CSV Data

Bubble charts are similar to scatter plots. The size of each dot (marker) is used to represent an additional dimension.

In the demo below, the bubble chart shows the relationship between median household income (x-axis) and test performance (y-axis) in 6 school districts in Connecticut. The size of data point (marker) corresponds to the number of students enrolled in the school district: bigger circles represent larger school districts.

Demo: https://handsondataviz.github.io/chartjs-templates/bubble-chart/index.html

Source and instructions: https://github.com//chartjs-templates/tree/master/bubble-chart

Tip: Use semi-transparent circles

Data points may obstruct each other. To avoid this, play with color transparency. For example, `rgba(160, 0, 0, 0.5)` is a semi-transparent red in RGBA color model. The `a` stands for `alpha`, and is a number between 0 and 1, where 1 is solid, and 0 is completely transparent. Using transparency, you will be able to see data points that are hidden behind bigger neighbors.

Going beyond three dimensions

To show more than three dimensions in bubble charts, one can:

- **color** each data point differently to show fourth dimension, eg use shades of red and green to show 5-year trend in test performance;
- use different **icons or glyphs** to display fifth dimension, eg circles for male students and squares for female students.

Remember not to overwhelm the viewer and communicate only the data that are necessary to prove or illustrate your idea.

Chapter 13

Leaflet Map Templates

While beginners appreciate the drag-and-drop map tools and tutorials described earlier in this book, Google My Maps and Carto, more advanced users may wish to customize their visualizations, add more complex data and interactivity, and control exactly how and where their work appears on the web. A more powerful and relatively easy-to-learn solution is to use code templates built with Leaflet https://leafletjs.com, an open-source library, which you can modify and host on GitHub, as described in this book.

13.0.0.1 Working with Leaflet

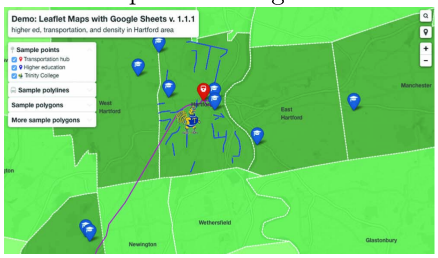
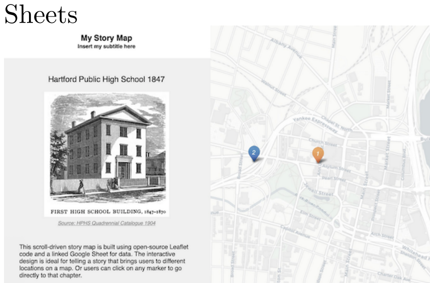
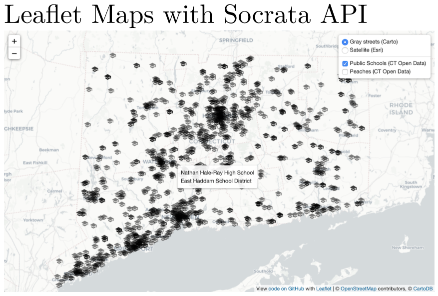
Pros:

- Open-source code, which anyone can freely use online, download, modify, or expand with plugins
- Easier for beginners to understand than some other map code libraries
- Compact code library (less than 40 KB) that runs on JavaScript in all modern web browsers

Cons:

- Must host your own code repositories to publish to the web (with a free service such as GitHub Pages)
- Must rely on open-source community of developers to maintain the core code or specific plugins

Leaflet Map Templates

Template	Best use and tutorial chapter
Leaflet Maps with Google Sheets 	Best to show points, polygons, polylines, or point table data. Upload your GeoJSON data and modify settings in linked Google Sheet (or CSV) and GitHub repo. Template with tutorial: Leaflet Maps with Google Sheets
Leaflet Storymaps with Google Sheets 	Create a scrolling narrative with points, text, images, and links. Template with tutorial: Leaflet Storymaps with Google Sheets
Leaflet Maps with Socrata API 	Create a Leaflet map powered by data from any Socrata data portal. Template with tutorial: Leaflet Maps with Socrata

Inside Leaflet code templates

The templates featured below vary from simple to complex, but all of them include three basic types of code:

- HTML: to structure content on the web (example: index.html)
- CSS, or Cascading Style Sheet: to shape how content appears on the web (example: style.css)
- JavaScript: to create the map and interactivity (example: script.js)

Also, these templates refer to other types of code:

- library: link to online instructions to complete routine tasks (examples: Leaflet, jQuery)

- basemap tiles: link to online background map (example: Carto Positron, a light-gray street map)
- data: content to appear on map, typically in CSV or GeoJSON format (examples: data.csv, data.geojson)

Leaflet Maps with Google Sheets template

Question: If you have moved beyond simple drag-and-drop point map tool, such as Google My Maps tutorials in this book, and want to create point and/or polygon and/or polyline maps, where should you go?

Answer: Copy and customize our open-source template for Leaflet Maps with Google Sheets. Control the map options display data that you upload to your Google Sheet and GitHub repository. No coding skills required, other than pasting one line of code to link your map with your sheet. Requires two free accounts: Google and GitHub.

Video and list of features

- Best to show points, polygons, and/or polylines, with table of points in map view
- Free and open-source code template, built on Leaflet and linked to Google Sheets
- Fork the code and host your live map on the web for free with GitHub Pages
- Geocode location data with US Census or Google, using script inside the Google Sheet
- Easy-to-modify data labels and map options in Google Sheet tabs or uploaded CSV files
- Upload your polygon and polyline GeoJSON files, and custom markers, to your GitHub repo
- Show multiple polygon layers, each with their own color legend and ranges (numerical or text)
- Responsive design resizes your maps to display inside most mobile devices

Try it

Explore the map or right-click to view full-screen map in a new tab

The map pulls the point data and settings from a linked Google Sheet, which you can explore below or right-click to view full-screen Sheet in a new tab

Part 1: Create and customize your map

In the first part of this tutorial, you will learn how to create your own copy of the Leaflet Maps with Google Sheets template, publish your Google Sheet, and paste its web address into your GitHub repo.

- A) Fork (copy) the code template and publish your version with GitHub Pages
- B) File > Make a Copy of Google Sheet template, Share, and File > Publish
- C) Paste your Google Sheet URL in two places in your GitHub repo
- D) Modify your map settings in the Options tab and test your live map

Part 2: Upload and display your map data

In the second part of this tutorial, you will learn how to geocode and customize your own point markers, and either hide or upload your own polygon and/or polyline GeoJSON data.

- E) Geocode locations and customize new markers in the Points tab
- F) Hide the polygon and polyline legends and default GeoJSON data
- G) Upload and display your own polygon GeoJSON data
- H) Upload and display your own polyline GeoJSON data
- I) Upload and display customized marker icons
- J) Optional: Save Google Sheets as CSV and upload to GitHub
- ** TO DO: second half video**

To solve problems, see Fix Common GitHub and Code Errors chapter in this book.

A) Fork (copy) the code template and publish your version with GitHub Pages

Before you begin, this tutorial assumes that you:

- have a free Google Drive account, and learned the File > Make a Copy in Google Sheets tutorial in this book

- have a free GitHub account, and understand concepts from the Modify and Host Code with GitHub chapter in this book

1) Right-click to open this GitHub code template in a new tab: https://github.com//leaflet-maps-with-google-sheets

2) In the upper-right corner of the code template, sign in to your free GitHub account

3) In the upper-right corner, click Fork to copy the template (also called a code repository, or repo) into your own account. The web address (URL) of the new copy in your account will follow this format:

```
https://github.com/USERNAME/leaflet-maps-with-google-sheets
```

Reminder: You can only fork a GitHub repo **one time**. If needed, see how to make a second copy in the Create a New Repo in GitHub chapter in this book.

4) In your new copy of the code repo, click on Settings, scroll down to the GitHub Pages area, select Master, and Save. This publishes your code to a live map on a public website that you control.

5) Scroll down to GitHub Pages section again, and copy the link to your published web site, which will follow this format:

```
https://USERNAME.github.io/leaflet-maps-with-google-sheets
```

Figure 13.1: Screencast: Fork

6) Scroll up to the top, and click on your repo name to go back to its main page.

7) At the top level of your repo main page, click on README.md, and click the pencil icon to edit this file, written in easy-to-read Markdown code.

8) Delete the link to the current live site, and paste in the link to YOUR site. Scroll down and Commit to save your edits.

9) On your repo main page, right-click the link to your live map to open in a new tab. **Be patient** during busy periods on GitHub, when your website may take up to 1 minute to appear the first time.

B) File > Make a Copy of Google Sheet template, Share, and File > Publish

1) Right-click to open this Google Sheets template in a new tab: https://docs.google.com/spreadsheets/d/1ZxvU8eGyuN9M8GxTU9acKVJv70iC3px_m3EVFsOHN9g

2) Sign into your Google account

3) File > Make a Copy of the Google Sheet template to your Google Drive

4) Click the blue Share button, click Advanced, click to change Private to Anyone with the link > Can View the Sheet. This will make your public data easier to view in your map.

Figure 13.2: Screencast: Share Google Sheet

5) File > Publish the Link to your Google Sheet to the public web, so the Leaflet map code can read it.

6) At the top of your browser, copy your Google Sheet web address or URL (which usually ends in `...XYZ/edit#gid=0`). Do NOT copy the published URL (which usually ends in `...XYZ/pubhtml`).

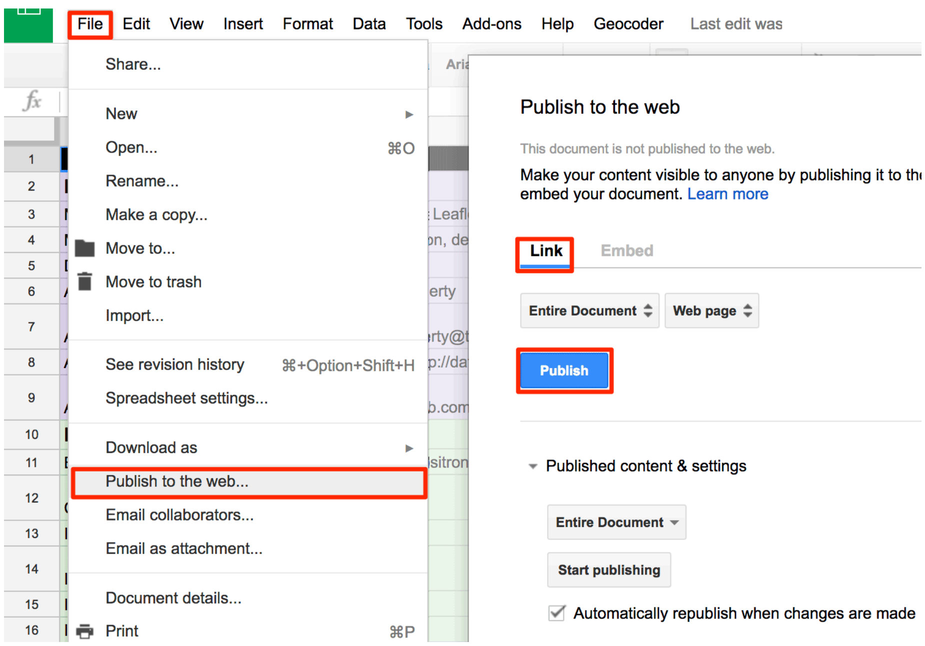


Figure 13.3: Screenshot: File > Publish the link to your Google Sheet

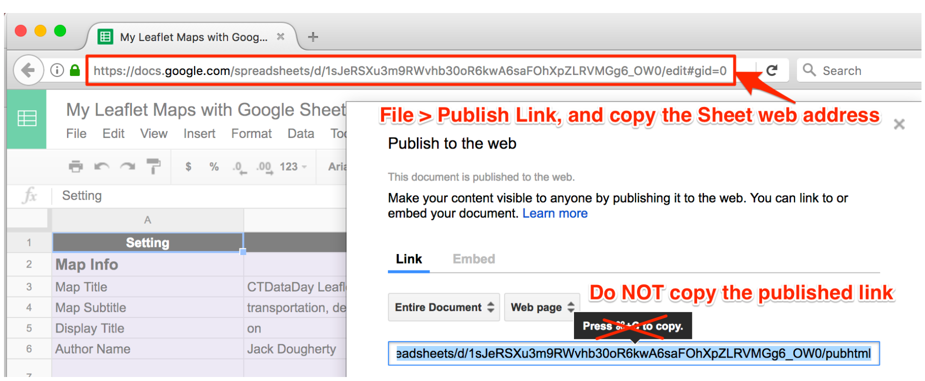


Figure 13.4: Screenshot: Copy the Google Sheet URL, not the Publish URL

C) Paste your Google Sheet URL in two places in your GitHub repo

1) First, connect your Google Sheet directly to your Leaflet Map code. In your Github code repo, click to open this file: `google-doc-url.js`
2) Click the pencil symbol to edit the file.
3) Paste your Google Sheet URL into the code to replace the current URL. Do not delete the single-quotation marks or semicolon.
4) Scroll to bottom of page and press Commit to save your changes. Now the Leaflet Map code can locate your published Google Sheet.

Figure 13.5: Screencast: Copy Google Sheet URL and paste into GitHub code

5) Next, let's paste your Google Sheet URL in a second place to keep track of it. Go to the README.md file in your GitHub repo, click to open and edit, and paste your Google Sheet web address to replace the existing link near the top. Commit to save your changes.

D) Modify your map settings in the Options tab and test your live map

In the top-level of your GitHub repo, test the new links to your map and your Google Sheet to make sure they work and point to your versions.

** TO DO - redo GIF **

In your linked Google Sheet, go to the Options Tab and modify these items:

1) Map Title – insert your own title
2) Map Subtitle – insert your own version

3) Author Name – insert your own name, or first name, or initials (will be public)
4) Author Email or Website – insert your own (will be public), or delete the current name to make it blank

Open the link to your live map in a new browser tab and refresh to see your changes.

E) Geocode locations and customize new markers in the Points tab

In your new map, our next goal is to add and modify the appearance of a new set of point markers, based on new addresses that you will enter and geocode.

In the Points tab of your Google Sheet:

1) Think of a simple data story that involves at least four geocodeable locations, divided into at least two groups. If you need an example, use this sample table of "Famous Places in New York City":

Group	Name	Location
Landmark	Empire State Building	350 5th Ave, New York, NY 10118
Landmark	Metropolitan Museum of Art	1000 5th Ave, New York, NY 10028
Transit	Grand Central Terminal	89 E 42nd St, New York, NY 10017
Transit	Penn Station	159 West 33rd Street, New York, NY 10120

2) Enter your Group, Name, and Location data into new rows below the current data.
3) Go to the Font Awesome Icons site, http://fontawesome.io/icons, scroll down to Search Icons, find your desired icon code name, and insert this into the Marker Icon (column B) of your Points sheet. For example, search for and insert the icon code "train" or "building" to display markers with either of these symbols in your map. (To upload your own customized marker, see section H further below.)
4) In Marker Color (column C), use the drop-down menu to select a marker color.
5) In Icon Color (column D), insert a color word (example: white) or hex code (example: #fff) to color the icon symbol inside your marker. Recommended: use white icon colors with dark marker colors.
6) Leave Custom Size (column E) blank.

7) Optional:

- In Image (column G), insert the URL (preferably https://, not http://) of a small-to-medium sized image on the web
- In Description (column G), insert text and/or a web link enclosed with an HTML a href tag with target set to blank

8) Do NOT delete or rename any column headers. However, you have the option to add new column headers to display in your map table.

9) Geocode your new data inside your Google Sheet by dragging your cursor to select 6 columns of data: Location - Latitude - Longitude - Found - Quality - Source

10) In the Geocoder menu that appears in this Google Sheet template, select one of the geocoding services. If one service cannot locate your data, try the other. Always inspect the accuracy of the Found column.

Open the link to your live map in a new browser tab and refresh to see your changes. If your new markers appear correctly, then delete the existing rows that came with this template.

F) Hide the polygon and polyline legends and default GeoJSON data

To show a simple point map, learn how to turn off and hide the polygon and polyline legend and default data that came with this template. (See how to add your own GeoJSON data in section G below.)

In your linked Google Sheet:

1) In the Options tab, Polyline Legend Position (cell B 35) – select Off to hide the legend

2) In the Polygons tab, Polygon Legend Position (cell B 4) – select Off to hide the legend

3) In the Polygons tab, Polygon GeoJSON URL (cell B 6) – delete contents to remove polygons

4) Go to the next tab, named Polygons1, in its drop-down menu, select Delete to remove sheet

5) In the Polylines tab, delete the entire row (rows 2 and 3) to remove the existing lines

Go to the browser tab with your new map, and refresh the page to see your changes.

Optional:

- in the Options tab, Display Table (cell B 29), turn off to hide the table in your map
- or modify the list of item in Table Columns (cell B 30) to change the display in your table

G) Upload and display your own polygon GeoJSON data

1) Prepare your polygon file in GeoJSON format. View or modify the GeoJSON file properties (such as name, data fields, etc.) with one of these tools:

- GeoJSON.io, http://geojson.io – Drag-and-drop your file, and select the Table tab to view or rename properties. See GeoJSON.io tutorial in this book, OR
- MapShaper, http://mapshaper.org – Drag-and-drop your file. To edit, see MapShaper tutorial in this book

2) In your GitHub repo, click to open the Geometry subfolder, then click Upload Files, drag-and-drop your geojson file, and Commit changes

** TO DO ** - turn on settings that you turned off in step F above

3) In your linked Google sheet, go to Polygons tab to adjust these settings:

- change Polygon GeoJSON URL (cell B 6) to match the pathname of the file you uploaded above
- change Polygon GeoJSON Name (cell B 5) to the title to be displayed for this polygon layer
- change Polygon Legend Title (cell B 3) for the title in the polygon legend box

4) To adjust the polygon legend colors and range, see the Polygon Data and Color Settings sections of the Polygon tab in Google Sheets.
5) The code supports multiple polygon layers, which you can add (or delete) by duplicating the Polygons tab. Name them Polygons1, Polygons2, etc.

- TO DO *
- Explain: To use both the automatic ColorBrewer Palette and manual colors, insert blanks (goes to automatic palette above), separated by semicolons.

H) Upload and display your own polyline GeoJSON data

Follow similar steps as described in the Polygon section above, but adjust settings in the Polylines tab of your linked Google Sheet.

I) Upload and display customized marker icons

** TO DO **

J) Optional: Save Google Sheets as CSV and upload to GitHub

If desired, you can save your table data with your code, rather than in a separate Google Sheet. Go to each Sheet tab and File > Save As in CSV format, with these file names:

- options.csv
- points.csv
- polygons.csv (also: polygons1.csv, polygons2.csv, etc.)
- polylines.csv
- notes.csv (or .txt)

Upload these files into the main level of your GitHub code repository, where the template will process them automatically.

Learn more

To solve problems, see Fix Common GitHub and Code Errors chapter in this book.

Leaflet Storymaps with Google Sheets and Scrolling Narrative

TODO: Add intro text

Try it

Explore the map or right-click to view full-screen map in a new tab

The map pulls the point data and settings from a linked Google Sheet, which you can explore below or right-click to view full-screen Sheet in a new tab

Features

- Show map points, text, images, and links with scrolling narrative
- Free and open-source code template, built on Leaflet and linked to Google Sheets
- Fork the code and host your live map on the web for free with GitHub Pages
- Geocode location data with US Census or Google, using script inside the Google Sheet
- Easy-to-modify data and map options in Google Sheet tabs or uploaded CSV files
- Responsive design resizes your maps to display inside most mobile devices

Create Your Own

- A) Fork (copy) the code template and publish your version with GitHub Pages
- B) File > Make a Copy of Google Sheet template, Share, and File > Publish
- C) Paste your Google Sheet URL in two places in your GitHub repo
- D) Modify your map settings in the Options tab and test your live map
- E) Geocode locations in the Points tab

To solve problems, see Fix Common GitHub and Code Errors in this book.

A) Fork (copy) the code template and publish your version with GitHub Pages

Before you begin, this tutorial assumes that you:

- have a free Google Drive account, and learned the File > Make a Copy in Google Sheets tutorial in this book
- have a free GitHub account, and understand concepts from the Modify and Host Code with GitHub chapter in this book

1) Right-click to open this GitHub code template in a new tab: https://github.com//leaflet-storymaps-with-google-sheets
2) In the upper-right corner of the code template, sign in to your free GitHub account

3) In the upper-right corner, click Fork to copy the template (also called a code repository, or repo) into your own account. The web address (URL) of the new copy in your account will follow this format:

```
https://github.com/USERNAME/leaflet-storymaps-with-google-sheets
```

Reminder: You can only fork a GitHub repo **one time**. If needed, see how to make a second copy in the Create a New Repo in GitHub chapter in this book.

4) In your new copy of the code repo, click on Settings, scroll down to the GitHub Pages area, select Master, and Save. This publishes your code to a live map on a public website that you control.
5) Scroll down to GitHub Pages section again, and copy the link to your published web site, which will follow this format:

```
https://USERNAME.github.io/leaflet-storymaps-with-google-sheets
```

6) Scroll up to the top, and click on your repo name to go back to its main page.
7) At the top level of your repo main page, click on README.md, and click the pencil icon to edit this file, written in easy-to-read Markdown code.
8) Delete the link to the current live site, and paste in the link to YOUR site. Scroll down and Commit to save your edits.
9) On your repo main page, right-click the link to your live map to open in a new tab. **Be patient** during busy periods on GitHub, when your website may take up to 1 minute to appear the first time.

B) File > Make a Copy of Google Sheet template, Share, and File > Publish

1) Right-click to open this Google Sheets template in a new tab: https://docs.google.com/spreadsheets/d/1AO6XHL_0JafWZF4KEejkdDNqfuZWUk3SlNlQ6MjlRFM/
2) Sign into your Google account
3) File > Make a Copy of the Google Sheet template to your Google Drive
4) Click the blue Share button, click Advanced, click to change Private to Anyone with the link > Can View the Sheet. This will make your public data easier to view in your map.
5) File > Publish the Link to your Google Sheet to the public web, so the Leaflet map code can read it.

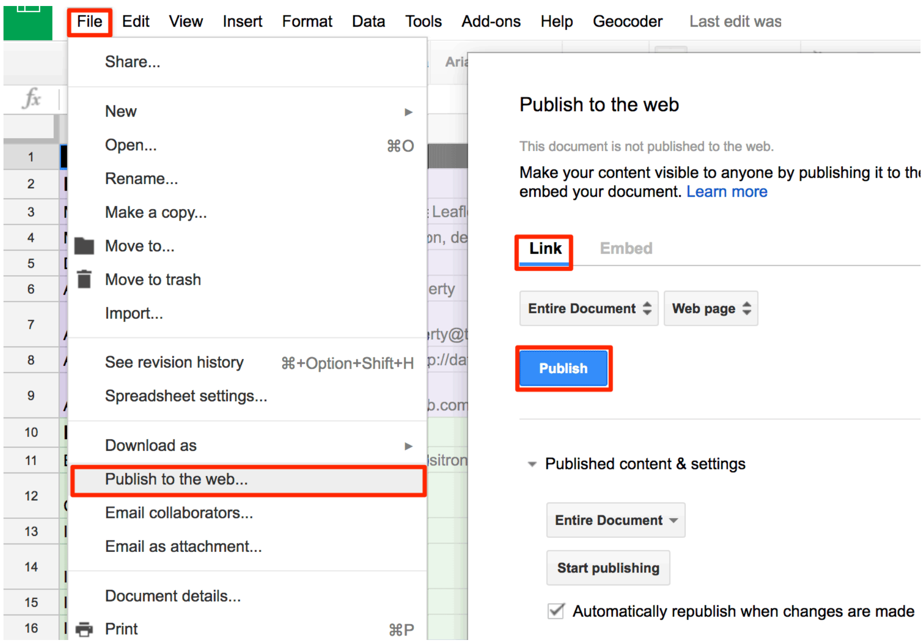


Figure 13.6: Screenshot: File > Publish the link to your Google Sheet

6) At the top of your browser, copy your Google Sheet web address or URL (which usually ends in `...XYZ/edit#gid=0`). Do NOT copy the published URL (which usually ends in `...XYZ/pubhtml`).

C) Paste your Google Sheet URL in two places in your GitHub repo

1) First, connect your Google Sheet directly to your Leaflet Map code. In your Github code repo, click to open this file: `google-doc-url.js`

2) Click the pencil symbol to edit the file.

3) Paste your Google Sheet URL into the code to replace the current URL. Do not delete the single-quotation marks or semicolon.

4) Scroll to bottom of page and press Commit to save your changes. Now the Leaflet Map code can locate your published Google Sheet.

5) Next, let's paste your Google Sheet URL in a second place to keep track of it. Go to the README.md file in your GitHub repo, click to open and edit, and paste your Google Sheet web address to replace the existing link near the top. Commit to save your changes.

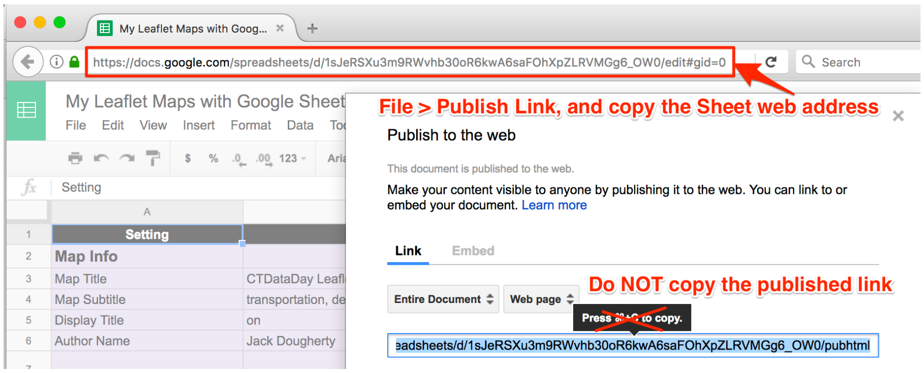


Figure 13.7: Screenshot: Copy the Google Sheet URL, not the Publish URL

D) Modify your map settings in the Options tab and test your live map

In the top-level of your GitHub repo, test the new links to your map and your Google Sheet to make sure they work and point to your versions.

** TO DO - redo GIF **

In your linked Google Sheet, go to the Options Tab and modify these items:

1) Map Title – insert your own title
2) Map Subtitle – insert your own version
3) Author Name – insert your own name, or first name, or initials (will be public)
4) Author Email or Website – insert your own (will be public), or delete the current name to make it blank

Open the link to your live map in a new browser tab and refresh to see your changes.

E) Geocode locations and customize new markers in the Points tab

In your new map, our next goal is to add and modify the appearance of a new set of point markers, based on new addresses that you will enter and geocode.

In the Points tab of your Google Sheet:

1) Do NOT delete or rename any column headers. However, you have the option to add new column headers to display in your map table.

2) Geocode your new data inside your Google Sheet by dragging your cursor to select 6 columns of data: Location - Latitude - Longitude - Found - Quality - Source
3) In the Geocoder menu that appears in this Google Sheet template, select one of the geocoding services. If one service cannot locate your data, try the other. Always inspect the accuracy of the Found column.

Open the link to your live map in a new browser tab and refresh to see your changes. If your new markers appear correctly, then delete the existing rows that came with this template.

TODO

Add documentation for new features added in 2020

Markers

I added a new column to the Chapter tab called "Marker". It has a dropdown with currently three options: Numerated (defaults to that, even if empty value), Plain (with no number), and No marker. The latter is what you want. It can be potentially extended to colours, types of markers, etc. https://github.com//leaflet-storymaps-with-google-sheets/blob/master/scripts/storymap.js#L121-L131

Overlay GeoJSONs

I added two columns, GeoJSON Overlay with the URL to the GeoJSON, and GeoJSON Feature Properties, which is CSS that defines style of features. List the styles separated by semicolon, and no quotation marks required. Eg fill-Color: orange; weight:2, opacity: 0.5, color: red, fillOpacity: 0.1 In the code, you will see two vertical lines: they mean "or". If the value of the left-most expression is not undefined, it uses it. If not, it keeps moving to the right until there is a value that is not an empty string. For example, https://github.com//leaflet-storymaps-with-google-sheets/blob/master/scripts/storymap.js#L310 color: feature.properties.COLOR || props.color || 'silver',

Will first attempt to extract the color from the COLOR property of each geoJson feature (useful for choropleth). If not found, it tries the GeoJSON Feature Properties "color". If that is not set, it uses silver. https://github.com//leaflet-storymaps-with-google-sheets/blob/master/scripts/storymap.js#L288-L316

Data in local CSV files

If googleDocURL variable does not exist (eg you delete the file) or is an empty string, it reads two spreadsheets: Options.csv and Chapters.csv from the /csv folder. Otherwise, it reads from the google sheet. https://github.com//leaflet-storymaps-with-google-sheets/blob/master/scripts/storymap.js#L13-L35 When data is read from a .CSV, it links that in the attribution (https://github.com//leaflet-storymaps-with-google-sheets/blob/master/scripts/storymap.js#L393-L396)

Learn more

To solve problems, see Fix Common GitHub and Code Errors chapter in this book.

Leaflet Maps with Socrata API Open Data

TODO: Decide whether to keep or not, and if so, add intro

Source: Current Class 1 - Class 4 Food Establishments, City of Hartford

Why pair Leaflet maps with Socrata data?

Leaflet, a friendly and flexible open-source code library for creating interactive web maps, plays nicely with Socrata, an open data platform used by several government agencies and organizations. Benefits of pairing Leaflet and Socrata:

- Although the Socrata data platform includes built-in visualization tools for anyone to create charts and maps, Leaflet gives you more control over your map design. Furthermore, Leaflet allows you to create maps that bring together data from both Socrata and non-Socrata sources.
- Socrata datasets include an API (application program interface) endpoint, in the form of a web address. This endpoint enables other computers to easily access the most recent data online, instead of a static version that was manually downloaded.
- Newer Socrata datasets that include locations (such as latitude and longitude coordinates) also provide endpoints in GeoJSON format. Since Leaflet maps easily process GeoJSON data, only a few lines of code are required.
- However, Socrata GeoJSON endpoints do not currently support "real-time" data, such as up-to-the-minute locations of public transportation, etc. In these cases, you may need to access data through a provider other than Socrata, most likely in a different format, which may require more coding skills.

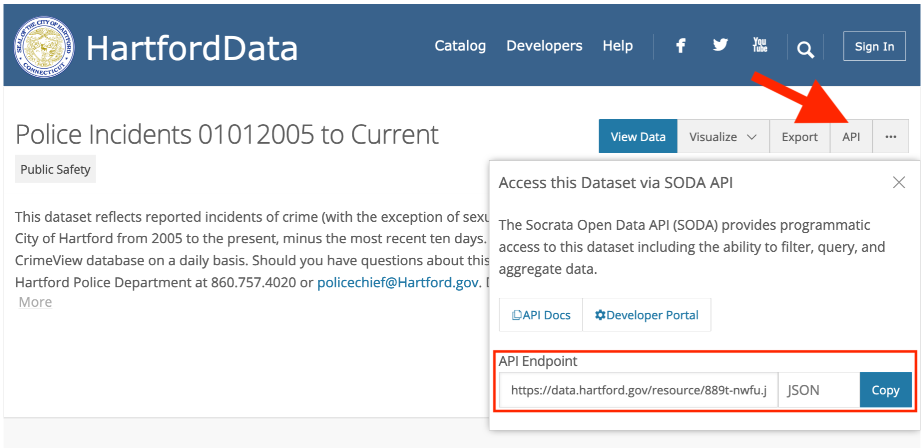


Figure 13.8: Police Incidents dataset on Hartford Open Data portal

About Socrata API endpoints

Go to any Socrata open data platform, find a dataset, and click the API tab. As an example, you can use City of Hartford's Police Incidents dataset.

Copy the API endpoint. The default version is JSON.

If you're new to APIs, test the endpoint by pasting it into your browser address line. Ideally you would see a formatted JSON view (use Chrome or Firefox for better results).

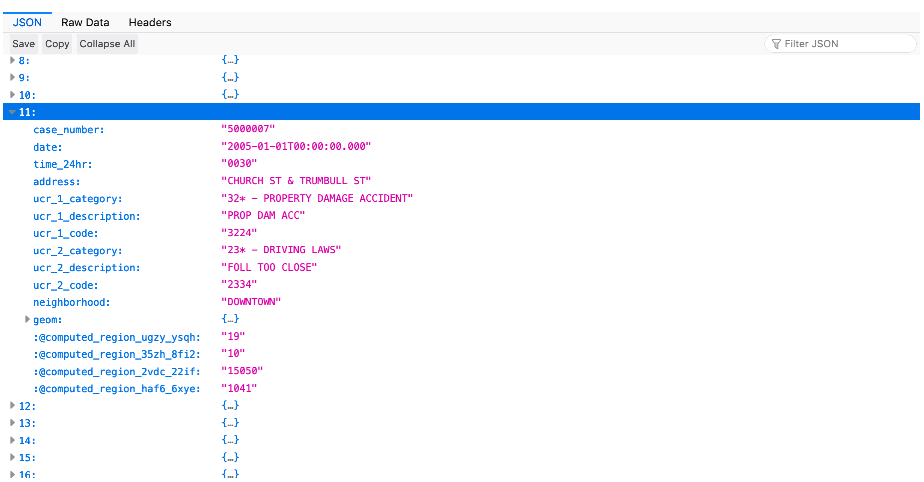


Figure 13.9: Formatted JSON example in Firefox

If your browser does not support JSON view, you will see the raw JSON stream only, like the one shown below.

Figure 13.10: Unformatted JSON example in Firefox

Test if this Socrata endpoint supports GeoJSON format by changing the extention in the API dropdown menu from `JSON` to `GeoJSON`. GeoJSON format works best with Leaflet because the coding is simpler.

If your endpoint supports GeoJSON format, your browser will display a data stream similar to the one below.

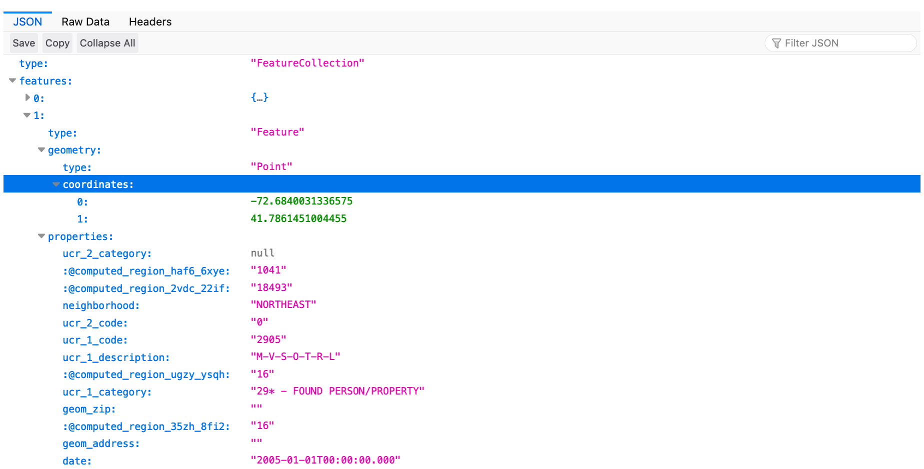


Figure 13.11: Formatted GeoJSON example in Firefox

If your Socrata endpoint only supports JSON format, but includes data columns with latitude and longitude, see other Leaflet examples further below.

Register for Socrata App Token

- Socrata requires developers to register for a free app token at https://opendata.socrata.com/signup

Demonstration Maps

GeoJSON endpoint with circle markers and tooltips

- map https://handsondataviz.github.io/leaflet-socrata/index.html
- code https://github.com//leaflet-socrata/index.html
- data https://data.hartford.gov/Public-Health/Current-Class-1-Class-4-Food-Establishments/xkvv-76v8
- note: location data appears as latitude and longitude coordinates in the `geom` column
- steps to create your own (MORE TODO HERE)
  - select API button, copy endpoint, and change suffix from .json to .geojson
  - copy this Leaflet map template, which includes this key section of code:
  - paste and explain the code

GeoJSON endpoint with simple data filter, default marker styling and pop-up info

- map https://handsondataviz.github.io/leaflet-socrata/index-geojson-filter
- code https://github.com//leaflet-socrata/
- data https://data.ct.gov/Environment-and-Natural-Resources/Agricultural-Commoditites-Grown-By-Farmer/y6p2-px98

Multiple Socrata datasets with Leaflet control layers legend

- map https://handsondataviz.github.io/leaflet-socrata/index-control-layers.html
- code https://github.com//leaflet-socrata/index-control-layers.html

Older JSON-only endpoint, with separate columns for latitude, longitude

- map https://handsondataviz.github.io/leaflet-socrata/index-json.html
- code https://github.com//leaflet-socrata/index-json.html
- data https://opendata.demo.socrata.com/Government/Kentucky-Farmers-Market-Map/3bfj-rqn7

Learn more

- https://dev.socrata.com/
- https://github.com/chriswhong/simpleSodaLeaflet

Thanks to

- Chris Metcalf https://github.com/chrismetcalf
- Tyler Klyeklamp https://data.ct.gov/

Pull Open Data into Leaflet Map with APIs

TODO: Decide whether to keep or not. Up to this point in the book, we've built charts and maps using static data that you have downloaded from other sites. But some open data repositories have APIs, or application program interfaces, which means the software that allows computers to communicate with one another. Below is a Leaflet Map template that uses APIs to pull in the most current data from three different open repository platforms: Socrata, Esri ArcGIS Online, and USGS.

Try it

Explore the map below or view full-screen version in a new tab

How it works

1) Go to the GitHub repo for the map above: https://github.com//leaflet-data-apis

2) Explore the code to see how different APIs work. For example, see the first map overlay, which pulls Connecticut School Directory data from the CT Open Data repository on a Socrata open data platform: https://data.ct.gov/resource/v4tt-nt9n

Figure 13.12: Screenshot: Sample API endpoint in Socrata open data repo

3) Inside the open data repo, look for an API button and copy the endpoint.

4) Paste the endpoint link into your browser, change the suffix from `.json` to `.geojson` and press return. In order to show the endpoint data as points on a map in this simple Leaflet template, the points must already be geocoded inside the open data repo, and the platform must support a GeoJSON endpoint. In your browser, one sign of success is a long stream of GeoJSON data like this:

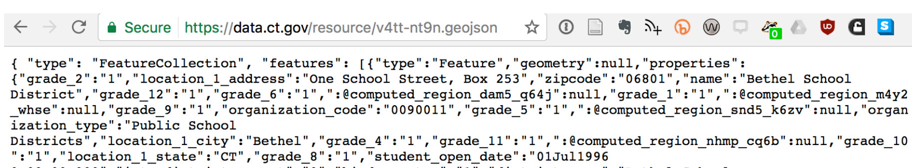


Figure 13.13: Screenshot: API endpoint with .geojson suffix in Chrome browser

5) In this section of the Leaflet map template, the code includes a jQuery function `$.getJSON` to call the open data endpoint in GeoJSON format: `https://data.ct.gov/resource/v4tt-nt9n.geojson`. It also requires a Socrata app token, and you can get your own token for free at: https://dev.socrata.com/register. Each geocoded school in the Socrata data repository is displayed as a blue circle, with data properties (such as: name) in a clickable pop-up.

```
// load open data from Socrata endpoint in GeoJSON format
// with simple marker styling: blue circles
// register your own Socrata app token at https://dev.socrata.com/register
// Connecticut School Directory, CT Open Data, https://data.ct.gov/resource/v4tt-nt9n
$.getJSON("https://data.ct.gov/resource/v4tt-nt9n.geojson?&$$app_token=QVVY3I72SVPbxBY
  var geoJsonLayer = L.geoJson(data, {
    pointToLayer: function( feature, latlng) {
      var circle = L.circleMarker(latlng, {
        radius: 6,
        fillColor: "blue",
        color: "blue",
        weight: 2,
        opacity: 1,
        fillOpacity: 0.7
      });
      circle.bindPopup(feature.properties.name + '<br>' + feature.properties.district_
      return circle;
    }
  }).addTo(map); // display by default
  controlLayers.addOverlay(geoJsonLayer, 'Public Schools (CT Open Data-Socrata)');
});
```

5) Fork a copy of this repo, play with the code, and try to insert GeoJSON endpoints from other open data repositories.

Leaflet Thematic Polygon Map with Clickable Info Window template

TODO: Decide whether to keep or not

Try it

View demo in new page

- https://handsondataviz.github.io/leaflet-map-polygon-click/

To Do

- Insert internal references to prior steps in this book. See the Edit and Host Code Templates section in this book.
- Requires a free GitHub account to host your own version on the web.

Create Your Own: Fork a copy of the code template on GitHub

- https://github.com//leaflet-map-polygon-click
- Remember, if you have already forked one copy, go to your GitHub repository Settings to rename it, or create a new GitHub repo and use GitHub Desktop to upload template Files

Obtain a polygon boundary map in GeoJSON format

- Find open data repositories to download maps in geojson and other formats
- If map is in shapefile or KML or other format, convert with http://geojson.io or http://mapshaper.org
- Import polygon map into http://mapshaper.org. In this example, map filename is: ct-towns-simple.geojson
  - See tutorial on Mapshaper.org to delete unwanted data columns or simplify file size
  - Export as CSV to create generic spreadsheet of polygon names. In this example, column header is "town"

Prepare your spreadsheet data and join with the polygon map

- Open CSV with any spreadsheet tool to view data column of polygon names.
- Download or prepare your new spreadsheet data in rows to match polygon names.
- Insert columns of data into the CSV sheet. Use VLOOKUP function if needed.
- Save CSV with new name. In this example: ct-towns.csv
- Import ct-towns.csv as second layer into MapShaper.org.
- Use the drop-down to select the polygon map (ct-towns-simple.geojson) as the active, displayed layer.
- Click the Console and enter command to join the CSV table to the GeoJSON polygon, where the matching data columns are both named "town"

  ```
  -join ct-towns.csv keys=town,town
  ```

- Export the newly joined map with a new filename in GeoJSON format
- Change the file suffix from .json to .geojson to avoid confusion. The new joined map data file is now named: ct-towns-density.geojson

Upload your map data and edit template in your GitHub repo

- The GitHub repo you created in the first step contains these files:
  - ct-towns-density-2010.csv (the spreadsheet joined into the polygon map)
  - ct-towns-density.geojson (the joined map data file)
  - index.html (the primary web page)
  - script.js (code to operate the map, to be modified below)
  - style.css (code that styles the map)
  - README.md (edit to insert a link to your own version)
  - LICENSE (terms of use for this free and open-source code)
- Upload your own map data geojson file
- Recommended: upload your own CSV spreadsheet file to
- In the script.js file, look for code comments labeled "Edit" to change references to geojson map data and its column headers, and also colors and ranges for the polygons and legend
- In GitHub, go to Branches and delete the existing "gh-pages" branch
- In GitHub, go to drop-down menu for Master branch, and type "gh-pages" to create new branch
- Content in the gh-pages branch will be hosted on the live web
- Edit the README.md link to point to your own gh-pages branch, in this format: `https://USERNAME.github.io/REPO-NAME/`

Leaflet Thematic Polygon Map with Hover Info Window template

TODO: Decide whether to keep or not

Try it

View demo in new page

- https://handsondataviz.github.io/leaflet-map-polygon-hover/

To Do

- Insert internal references to prior steps in this book. See the Edit and Host Code Templates section in this book.
- Requires a free GitHub account to host your own version on the web.

Create Your Own: Fork a copy of the code template on GitHub

- https://github.com//leaflet-map-polygon-hover/
- Remember, if you have already forked one copy, go to your GitHub repository Settings to rename it, or create a new GitHub repo and use GitHub Desktop to upload template Files

TO DO - describe all steps, which are similar to click version

Leaflet Thematic Polygon Map with Multi-Year Tabs template

TODO: decide whether to keep or not

Try it

View demo in new page

- https://handsondataviz.github.io/leaflet-map-polygon-tabs/

** To Do **

- Insert internal references to prior steps in this book. See the Edit and Host Code Templates section in this book.
- Requires a free GitHub account to host your own version on the web.
- describe all steps, which are similar to the prior chapter

Create Your Own: Fork a copy of the code template on GitHub

- https://github.com//leaflet-map-polygon-tabs/
- Remember, if you have already forked one copy, go to your GitHub repository Settings to rename it, or create a new GitHub repo and use GitHub Desktop to upload template Files

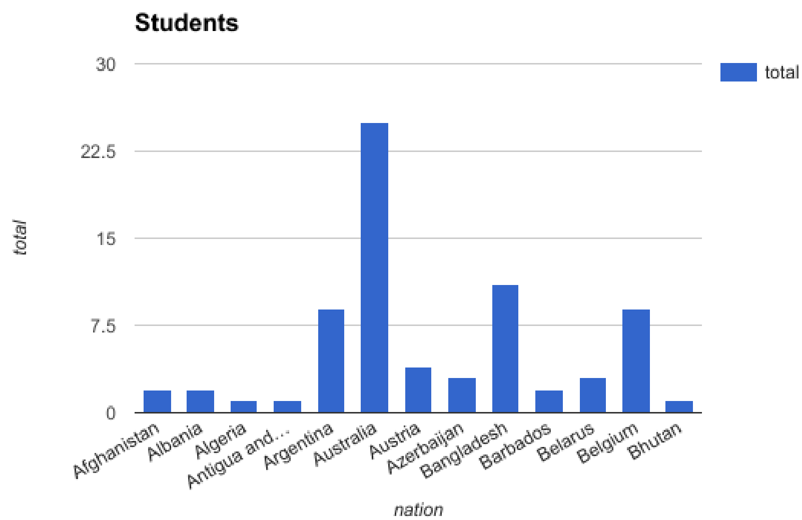
Chapter 14

Peer Review Samples

ONLY FOR WEB EDITION: The next pages include partial-credit and full-credit samples for peer review in the Data Visualization for All edX course.

Section 2 Chart 1 Peer Review Sample

Students in the Data Visualization for All course come from several different countries, including Australia, Bangladesh, and Belgium.

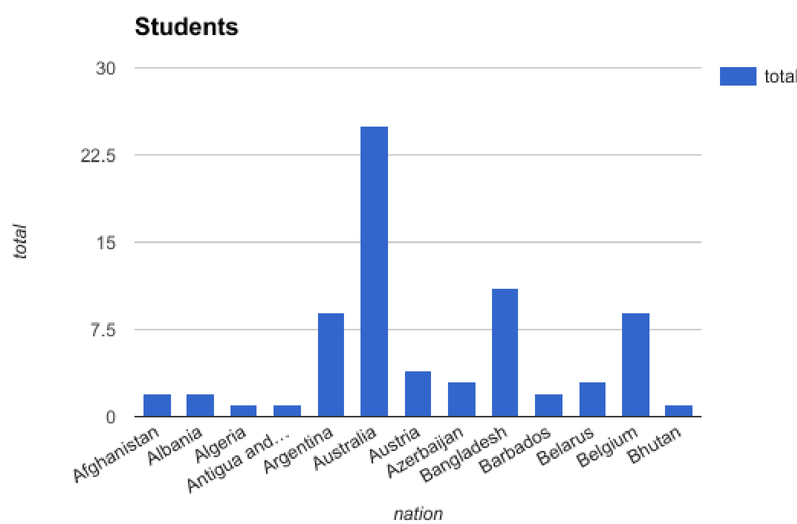


Evaluate

1. Story: Did the author clearly tell a meaningful story about the data, with text and visuals?
2. Chart Type: Did the author choose a chart type that best matches their data story?
3. Embed: Did the author embed an interactive chart into the web page?
4. Good Design: Did the author follow principles of good chart design?

Section 2 Chart 1 Peer Review Sample with Notes

Students in the Data Visualization for All course come from several different countries, including Australia, Bangladesh, and Belgium.



Evaluate

1. Story: Did the author clearly tell a meaningful story about the data, with text and visuals?

- No, this simple statement that students come from "several different countries" is not a very meaningful story.

2. Chart Type: Did the author choose a chart type that best matches their data story?

- No. Although a vertical column chart is a good start, a horizontal bar chart would be a better match for these long labels.

3. Embed: Did the author embed an interactive chart into the web page?

- No, when you try to float your cursor over the chart, it is a static image, not an interactive visualization.

4. Good Design: Did the author follow principles of good chart design?

- No, the chart ignores several design principles, such as: - Failure to sort data into a meaningful order - Failure to declutter the chart by removing the unnecessary legend

Section 2 Chart 2 Peer Review Sample

Nations with the highest percentage of female students enrolled in Data Visualization for All are the Ukraine (51 percent) and Turkey (47 percent), based on preliminary data for those with high enrollment levels (25 or more students).

View the preliminary data for 21 Feb 2017 from http://handsondataviz.org

Evaluate

1. Story: Did the author clearly tell a meaningful story about the data, with text and visuals?
2. Chart Type: Did the author choose a chart type that best matches their data story?
3. Embed: Did the author embed an interactive chart into the web page?
4. Good Design: Did the author follow principles of good chart design?

Section 2 Chart 2 Peer Review Sample with Notes

Nations with the highest percentage of female students enrolled in Data Visualization for All are the Ukraine (51 percent) and Turkey (47 percent), based on preliminary data for those with high enrollment levels (25 or more students).

View the preliminary data for 21 Feb 2017 from http://handsondataviz.org

Evaluate

1. Story: Did the author clearly tell a meaningful story about the data, with text and visuals?

- Yes, this insight on gender differences in student enrollments across nations is a meaningful story.

2. Chart Type: Did the author choose a chart type that best matches their data story?

- Yes, this stacked horizontal bar chart is a good match for showing part-to-whole relationships (gender by percentage) between different nations.

3. Embed: Did the author embed an interactive chart into the web page?

- Yes, when you float your cursor over it, the interactive chart tooltip shows data labels and values.

4. Good Design: Did the author follow principles of good chart design?

- Yes, the chart demonstrates good design principles, such as:
  - Sorting data into a meaningful order
  - Using color contrast (blue vs grays) to highlight percentages of female students

Section 3 Peer Review Sample 1

My Leaflet map

My Highcharts scatter chart

Evaluate

1. Leaflet map and title: Did the author embed an interactive Leaflet map with a new title?
2. Leaflet map layers: Did the author add controls that toggle on/off different map layers?
3. Leaflet point markers: Did the author upload a new set of markers, with pop-ups that show titles for each point?
4. Highcharts scatter chart: Did the author embed an interactive Highcharts scatter chart with a new title and axis labels?

5. Highcharts data tooltips: Did the author upload a new set of data, with tooltips that show labels and details for each point?
6. Additional comments for the author. What works well? What could be improved?

Section 3 Peer Review Sample 1 with Notes

My Leaflet map

My Highcharts scatter chart

Evaluate

1. Leaflet map and title: Did the author embed an interactive Leaflet map with a new title?

- No, the map title is the same as the original template, and is not new.

2. Leaflet map layers: Did the author add controls that toggle on/off different map layers?

- No, the map does not contain layer controls.

3. Leaflet point markers: Did the author upload a new set of markers, with pop-ups that show titles for each point?

- No, the map only contains one point, and the author did not upload a new set of points.

4. Highcharts scatter chart: Did the author embed an interactive Highcharts scatter chart with a new title and axis labels?

- No, the chart title and axis labels are the same as the original template, and are not new.

5. Highcharts data tooltips: Did the author upload a new set of data, with tooltips that show labels and details for each point?

- No, the author did not upload a new set of data points.

6. Additional comments for the author. What works well? What could be improved?

Section 3 Peer Review Sample 2

My Leaflet map

My Highcharts scatter chart

Evaluate

1. Leaflet map and title: Did the author embed an interactive Leaflet map with a new title?
2. Leaflet map layers: Did the author add controls that toggle on/off different map layers?
3. Leaflet point markers: Did the author upload a new set of markers, with pop-ups that show titles for each point?
4. Highcharts scatter chart: Did the author embed an interactive Highcharts scatter chart with a new title and axis labels?
5. Highcharts data tooltips: Did the author upload a new set of data, with tooltips that show labels and details for each point?
6. Additional comments for the author. What works well? What could be improved?

Section 3 Peer Review Sample 2 with Notes

My Leaflet map

My Highcharts scatter chart

Evaluate

1. Leaflet map and title: Did the author embed an interactive Leaflet map with a new title?

- Yes, the title in this map has changed from the original template

2. Leaflet map layers: Did the author add controls that toggle on/off different map layers?

- Yes, this map contains map layer controls.

3. Leaflet point markers: Did the author upload a new set of markers, with pop-ups that show titles for each point?

- Yes, this map contains new points that were added to the original template.

4. Highcharts scatter chart: Did the author embed an interactive Highcharts scatter chart with a new title and axis labels?

- Yes, this chart contains a new title and axis labels.

5. Highcharts data tooltips: Did the author upload a new set of data, with tooltips that show labels and details for each point?

- Yes, this chart contains a new set of data points that were uploaded to the original.

6. Additional comments for the author. What works well? What could be improved?

Chapter 15

Publishing with Bookdown

This open-access book is built with free-to-use, open-source tools—primarily Bookdown, GitHub, and Zotero—and this chapter explains how, so that readers may do it themselves and share their knowledge to improve the process. In addition to our notes below, see also Yihui Xie's more comprehensive Bookdown guide.[1]

Our broad goal is an efficient workflow to compose one document in the easy-to-write Markdown format that Bookdown generates into multiple book products: an HTML web edition to read online, a PDF print edition for traditional book publishing, a Microsoft Word edition for editors who request it for copyediting, and option for other formats as desired.

Since Bookdown is an R code package, we composed the book manuscript in R-flavored Markdown, with one file (.Rmd) for each chapter. We use Bookdown to build these files in its GitBook style as a set of static HTML pages, which we upload to our GitHub repository. Readers can view the open-access web edition of the book at our custom domain: https://HandsOnDataViz. Also, we use Bookdown to build additional book outputs (PDF, MS Word, Markdown) and upload these to the `docs` folder of our GitHub repository, so that our O'Reilly Media editor may download and comment on the manuscript as we revise. Finally, we also have the option to use Pandoc alone to convert the full-book Markdown file (.md) into an AsciiDoc file (.asciidoc) for easier importing into the O'Reilly Atlas platform. See some caveats and workarounds below.

File Organization and Headers

We organized the GitHub repository for this book as a set of .Rmd files, one for each chapter. As co-authors, we are careful to work on different chapters of the

[1] Yihui Xie, *Bookdown: Authoring Books and Technical Documents with R Markdown*, 2018, https://bookdown.org/yihui/bookdown/

book, and to regularly push our commits to the repo. Only one of us regularly builds the book with Bookdown to avoid code merge conflicts.

Bookdown assigns a default ID to each header, which can be used for cross-references. The default ID for `# Topic` is `{#topic}`, and the default ID for `## Section Name` is `{#section-name}`, where spaces are replaced by dashes. But we do *not* rely on default IDs because they might change due to editing or contain duplicates across the book.

Instead, we *manually assign a unique ID* to each first- and second-level header in the following way. Note that the `{-}` symbol, used alone or in combination *with a space* and a unique ID, prevents auto-numbering in the second- thru fourth-level headers:

```
# Top-level chapter title {#unique-name}
## Second-level section title {- #unique-name}
### Third-level subhead {-}
#### Fourth-level subhead {-}
```

Also, we match the unique ID keyword to the file name for top-level chapters this way: `01-keyword.Rmd` to keep our work organized. Unique names should contain only *alphanumeric* characters (a-z, A-Z, 0-9) or dashes (-).

In the Bookdown `index.Rmd` for the HTML book output and the PDF output, the `toc_depth: 2` setting displays chapter and section headers down to the second level in the Table of Contents.

The `split_by: section` setting divides the HTML pages at the second-level header, which creates shorter web pages with reduced scrolling for readers. For each web page, the unique ID becomes the file name, and is stored in the `docs` subfolder.

The `number_sections` setting is true for the HTML and PDF editions, and given the `toc_depth: 2`, this means that they will display two-level chapter-section numbering (1.1, 1.2, etc.) in the Table of Contents. Note that `number_sections` must be true to display Figure and Table numbers in x.x format, which is desired for this book. See relevant settings in this excerpt from `index.Rmd`:

```
output:
  bookdown::gitbook:
    ...
    toc_depth: 2
    split_by: section
    number_sections: true
    split_bib: true
    ...
```

```
bookdown::pdf_book:
  toc_depth: 2
  number_sections: true
```

Note that chapter and section numbering do *not* appear automatically in the MS Word output unless you supply a reference.docx file, as described below:

- https://bookdown.org/yihui/rmarkdown/word-document.html
- https://stackoverflow.com/questions/52924766/numbering-and-referring-sections-in-bookdown
- https://stackoverflow.com/questions/50609212/caption-styles-for-word-document2-in-bookdown

In the `_bookdown.yml` settings, all book outputs are built into the `docs` subfolder of our GitHub repo, as shown in this excerpt:

```
output_dir: "docs"
book_filename: "HandsOnDataViz"
language:
  label:
    fig: "Figure "
chapter_name: "Chapter "
```

In our GitHub repo, we set GitHub Pages to publish to the web using `master/docs`, which means that visitors can browse the source files at the root level, and view the HTML web pages hosted in the `docs` subfolder. We use the GitHub Pages custom domain setting so that the HTML edition is available at https://HandsOnDataViz.org.

The `docs` subfolder also may contain the following items, which are *not* generated by Bookdown and need to be manually created:

- `CNAME` file for the custom domain, generated by GitHub Pages.
- `.nojekyll` invisible empty file to ensure speedy processing of HTML files by GitHub Pages.
- `404.html` custom file to redirects any mistaken web addresses under the domain back to the `index.html` page.

One more option is to copy the Google Analytics code for the web book, paste it into an HTML file in the book repo, and include this reference in the `index.Rmd` code:

```
output:
  bookdown::gitbook:
  ...
  includes:
    in_header: google-analytics.html
```

Style Guide

View the source code to better understand how this page was composed at: https://github.com/HandsOnDataViz/book/blob/master/01-bookdown.Rmd

This book is composed in R-flavored Markdown (.Rmd), and each paragraph begins on a separate line. O'Reilly style guide prefers *italics* rather than bold. Use single back tics to display a monospaced `code` word.

Insert an embedded link to O'Reilly. This appears as a colored clickable link in HTML and Word editions, and a non-colored but clickable link in the PDF edition. According to O'Reilly Atlas documentation, the AsciiDoc version should automatically unfurl for the printed edition.

For lists, always insert a blank line *before* the items, unless they appear directly after hashtag header.

- unordered
- list

1. ordered
2. list

Dashes:

- Use a hyphen (1 dash) for hyphenated words, such as two-thirds or dog-friendly hotel.
- Use an en-dash (2 dashes) for ranges, such as the May–September magazine issue.
- Use an em-dash (3 dashes) to insert an additional thought—like this—in a sentence.

Insert `TODO` to note items to finish or review with co-author or editor.

Insert three back tics to insert a code block. Check character line length limits in O'Reilly style guide:

```
<link rel="stylesheet" href="https://unpkg.com/leaflet@1.6.0/dist/leaflet.css" />
<script src="https://unpkg.com/leaflet@1.6.0/dist/leaflet.js"></script>
```

Conditional Formatting

Conditional formatting offers the option to display text or images in some editions, but not other editions. Options:

1. Insert a HTML code comment `<!-- Comment -->` in the .Rmd file to hide a few lines of text. This appears as commented-out text in the HTML and .md formats, is not displayed in the HTML browser, and does not appear in any way in the PDF, MS Word or AsciiDoc formats.

Demo:

2. R package function `is_[html/latex]_output` allows conditional output for different book products, such as text that should appear in the HTML edition but not the PDF edition, or vice versa.

Demos:

This line appears in the PDF and Word versions, and is commented-out in the HTML and Markdown and AsciiDoc versions.

TODO: Create conditional formatting that displays *only* in the HTML edition, and allows the inclusion of R code-chunks to conditionally display images. See links for more complex conditional formatting:

- https://stackoverflow.com/questions/56808355/how-to-conditionally-process-sections-in-rmarkdown
- https://bookdown.org/yihui/rmarkdown-cookbook/latex-html.html
- https://blog.earo.me/2019/10/26/reduce-frictions-rmd/
- https://stackoverflow.com/questions/53861244/html-specific-section-in-bookdown
- https://stackoverflow.com/questions/41084020/add-a-html-block-above-each-chapter-header
- https://stackoverflow.com/questions/45360998/code-folding-in-bookdown

3. Option to customize the `style.css` code for the HTML book.

4. Option to add headers, footers, preambles to the HTML or LaTeX versions.

5. Build different versions of the HTML and LaTeX (PDF) books using different chapters by listing them in order in the `_bookdown.yml` file:

```
rmd_files:
  html: ["index.Rmd", "abstract.Rmd", "intro.Rmd"]
  latex: ["abstract.Rmd", "intro.Rmd"]
```

Cross-references

In order to cross-reference in Bookdown, assign a unique name or R code-chunk label to each chapter, section, figure, and table. Unique names and labels should contain only *alphanumeric* characters (a-z, A-Z, 0-9) or dashes (-).

To cross-reference any *chapter or section*, and allow readers to jump there, use a HTML link with the unique name, such as `index.html` or `style-guide.html`. Demos:

- See Introduction
- See "Style Guide" in Chapter x.

Contrary to the Bookdown manual, *avoid* using Bookdown unique ID links to cross-reference chapters or sections, because these create extraneous and imprecise URLs, such as this bad example.

To cross-reference figures and tables, and display their auto-number and allow readers to jump there, write a call-out with a Bookdown reference to a code-chunk label, such as `See Figure \@ref(fig:sample-map)` or `See Table \@ref(tab:left-table)`. Demos:

- See Figure 15.1.
- See Table 15.2.

Cross-reference interactivity varies by output:

- In HTML, all cross-refs are clickable.
- In PDF, all cross-refs are clickable (except chapter-level HTML links).
- In Word, no cross-refs are clickable (unless this varies with reference.docx).
- TBA with Markdown (.md) and AsciiDoc.

When writing cross-references in the text, the O'Reilly Style Guide prefers live cross references (e.g., "see Figure 2-1"), but if not feasible, use "preceding" or "following" because physical placement of elements may vary across print and digital formats. *Avoid* using "above" or "below."

Images

Create high-resolution color static images in .jpg or .png format, and animated .gif files, and save them into the `images` subfolder by chapter. Make sure that color images can be rendered into grayscale by the publisher for the print book. Write file names in lowercase with dashes (not spaces) and begin with keyword of relevant section to keep related images grouped together. Despite being in separate folders, avoid duplicate image file names across the book. Avoid numbering images since they may not match the final sequence. Add `-original` to the end of the file name prior to resizing or adding more text or artwork.

Use Photoshop or a similar photo-editing tool (*not* Preview) to add any additional text or artwork if desired. Try to maintain a high resolution (300 dpi) and reduce size if desired to fit into the HTML book (measured in pixels) and PDF book (measured in inches). Save into the same folder with the same file name, minus `-original`, like this:

```
images/05-chart/design-no-junk-original.png
images/05-chart/design-no-junk.png
```

When inserting image filenames into the text, use the version minus `-original`. If creating images to appear as the same size in sequence, add a code-comment with the image width, height, and resolution as a reminder to make others match up, like this:

```
<!-- Images below are 200x200 at 300 resolution -->
```

In this book, use *Markdown formatting only for images that appear inside tables* or *do not require captions or figure numbering*, and leave the caption field blank, like this example:

Co-Authors	About Us
	About Jack Dougherty
	About Ilya Ilyankou

Although Markdown formatting offers a simple syntax that easily converts into other formats with Bookdown/Pandoc, there is no auto-numbering in the HTML edition, while auto-numbering appears in the PDF edition, and numbered figures are required by the publisher. Furthermore, Markdown formatting does not allow conditional output.

For these reasons, this book *primarily uses R code-chunk formatting for images*. The syntax is more complex but supports auto-numbering in HTML and PDF, and conditional output for interactive and static images. Note that R code-chunk images do *not* easily convert with Pandoc from Markdown to AsciiDoc, but "Figure x Caption" appears as a placeholder.

Auto-numbering appears in `Figure x.x` format in HTML and PDF, but `Figure x` format in MS Word. TODO: Check if Word formatting can be changed with reference.docx.

Note that images in PDF output will "float" by design and may appear before or after the desired page, so always add a cross-reference call-out.

Write R code-chunk labels that follow the image file name, and avoid duplicate labels across the book:

```
ref:design-no-junk

images/05-chart/design-no-junk.png
```

Do not insert spaces inside the `ref:chunk-label` for the caption, but add a blank line to separate it from the code-chunk. After the code-chunk, add another blank line.

Demo: R code-chunk for static image

...as shown in Figure 15.1.

Figure 15.1: Caption here. Markdown embedded links are acceptable.

R code-chunks allow more complex conditional formatting, where an interactive map or animated GIF or YouTube video clip appears in the web version, and a static image with an embedded link appears in the PDF and MS Word outputs. Also note the option to change the default iframe height (400px) and width with settings in the `custom-scripts.html` file, with a code comment reminder.

Demo: R code-chunk for HTML iframe and static image

...as shown in Figure 15.2.

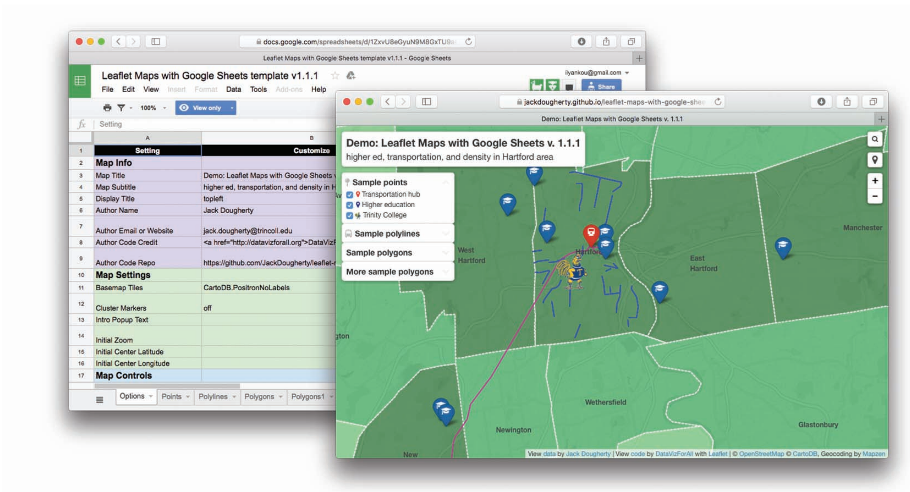


Figure 15.2: Caption here, and add embedded link to explore the full-screen interactive map.

Demo: R code-chunk for GIF animation and static image

TODO: FIX to make sizing appear consistently across browsers (FFox vs Chrome).

. . . as shown in Figure 15.3.

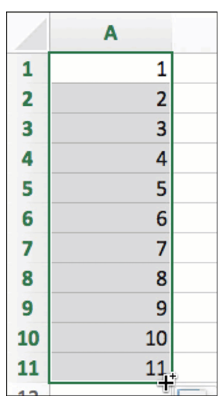


Figure 15.3: Caption here, with embedded link to GitHub repo, not GitHub Pages animated GIF.

Demo: R code-chunk for Youtube video and static image

Be sure to use the *embed* link from the YouTube *share* button.

. . . as shown in the video 15.4.

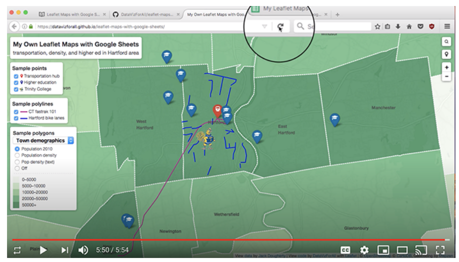


Figure 15.4: Caption here, with embedded link to the YouTube video.

This Bookdown `index.Rmd` file includes an R code-chunk setting immediately after the first header, which displays each code-chunk image without a code echo. Read more about this feature and related options in this Bookdown chapter.

```
{r setup, include=FALSE}
knitr::opts_chunk$set(echo = FALSE)
```

Tables

Create tables in Markdown format, since it produces good output for HTML, PDF, Word, and Markdown. Use a tool such as Tables Generator to import significant table data in CSV format, format the column alignment as desired, and press Generate button to create table in Markdown format. For significant table data, save the CSV version in a GitHub repo for potential later use.

Add the Markdown table code shown below to auto-number (Table x) in HTML, PDF, Word.

. . . as shown in Table 15.2.

Table 15.2: Left-justify content, remember blank Line

Much Much Longer Header	Short Header	Short Header
Left-justify text content with left-colons	Less	Here
Use more hyphens to grant more space to some columns	Less	Here

Table 15.3: Right-justify content, remember blank line

Header1	Header2	Header3
123	456	789
Right-justify	numerical content	with right-colons
Use equal hyphens	to make equal space	for all columns

Workaround for Markdown-to-AsciiDoc: Currently, our attempt to use Pandoc to directly convert a Bookdown-generated Markdown file to AsciiDoc fails because Bookdown creates the .md file with tables in .html format, not Markdown. Our workaround is to paste the individual Markdown-formatted tables directly from the .Rmd into the large .md file prior to converting with Pandoc to AsciiDoc.

Notes and Bibliography

This book displays endnotes for each chapter in the HTML book, and footnotes at the bottom of pages for the PDF and MS Word books, followed by an alphabetized bibliography of all references cited on the last page. The notes and bibliography also appear in the full-book Markdown file.

To create notes, insert citation keys in the text, such as `@huffHowLieStatistics1954`, which are generated by Zotero bibliographic database with the Better BibTex extension, and export these in the *Better BibLaTeX* format into the `dataviz.bib` in the book repo. The repo also contains `.csl` file to generate the notes and bibliography in a specific Chicago-style format, downloaded from the Zotero Styles Repository. These instructions are referenced in the `index.Rmd` file for both the HTML and PDF formats, as shown in these excerpts:

```
bibliography: dataviz.bib
citation-style: chicago-fullnote-bibliography.csl
...
output:
  bookdown::gitbook:
    ...
    pandoc_args: [ "--csl", "chicago-fullnote-bibliography.csl" ]

  bookdown::pdf_book:
    ...
    citation_package: none
    pandoc_args: [ "--csl", "chicago-fullnote-bibliography.csl" ]
```

Here's a text-only note, with no Zotero citation.[2]

[2] This is a note, with no bibliographic reference.

To create a note with citations only, separate Zotero/BibTeX citation keys with semi-colons:[3]

Since notes also may include text and punctuation in Markdown syntax, always insert a caret symbol prior to the brackets to demarcate a note:[4]

Note that the `chicago-fullnote-bibliography.csl` format automatically shortens the note after it its first reference.

Pandoc Conversion

- Download Pandoc
- Set Bookdown to build the book as one large Markdown file (docs folder, suffix .md)
- Use command line to navigate to subfolder with `pwd` and `cd`.
- Convert with: `pandoc handsondataviz.md --from markdown --to asciidoc --standalone --output handsondataviz.asciidoc`
- Confirm if AsciiDoc file matches O'Reilly Atlas import style.

[3] Darrell Huff, *How to Lie with Statistics* (W. W. Norton & Company, 1954–2010), http://books.google.com/books?isbn=0393070875; Mark S. Monmonier, *How to Lie with Maps*, 2nd ed. (University of Chicago Press, 1996), http://books.google.com/books?isbn=0226534219

[4] Compare how "lying" is justified by Huff, *How to Lie with Statistics*, pp. 10-11 and Monmonier, *How to Lie with Maps*, pp. 11-12.

Chapter 16

References

Huff, Darrell. *How to Lie with Statistics*. W. W. Norton & Company, 1954–2010. http://books.google.com/books?isbn=0393070875.

Monmonier, Mark S. *How to Lie with Maps*. 2nd ed. University of Chicago Press, 1996. http://books.google.com/books?isbn=0226534219.

Xie, Yihui. *Bookdown: Authoring Books and Technical Documents with R Markdown*, 2018. https://bookdown.org/yihui/bookdown/.