Web Scraping
Extracting Data from Websites

Eric Hettler

Wisconsin Department of Natural Resources

Web scraping: background

General concept: Extract data from a website so it is more usable.
Also known as “web data extraction”

Basic web scraping: Copy/Paste from websites

Advanced web scraping: Automate programs to extract data from websites

Resource: https://www.zyte.com/learn/what-is-web-scraping/#What-is-web-scraping?

Web scraping: Applications

Real Estate

REDFIN

Zillow®

REALTOR® | MLS MULTIPLE LISTING SERVICE

Consumer Sentiment

e-commerce

amazon.com

Google Shopping

Walmart+

Job Listings

indeed

MONSTER

LinkedIn

Travel

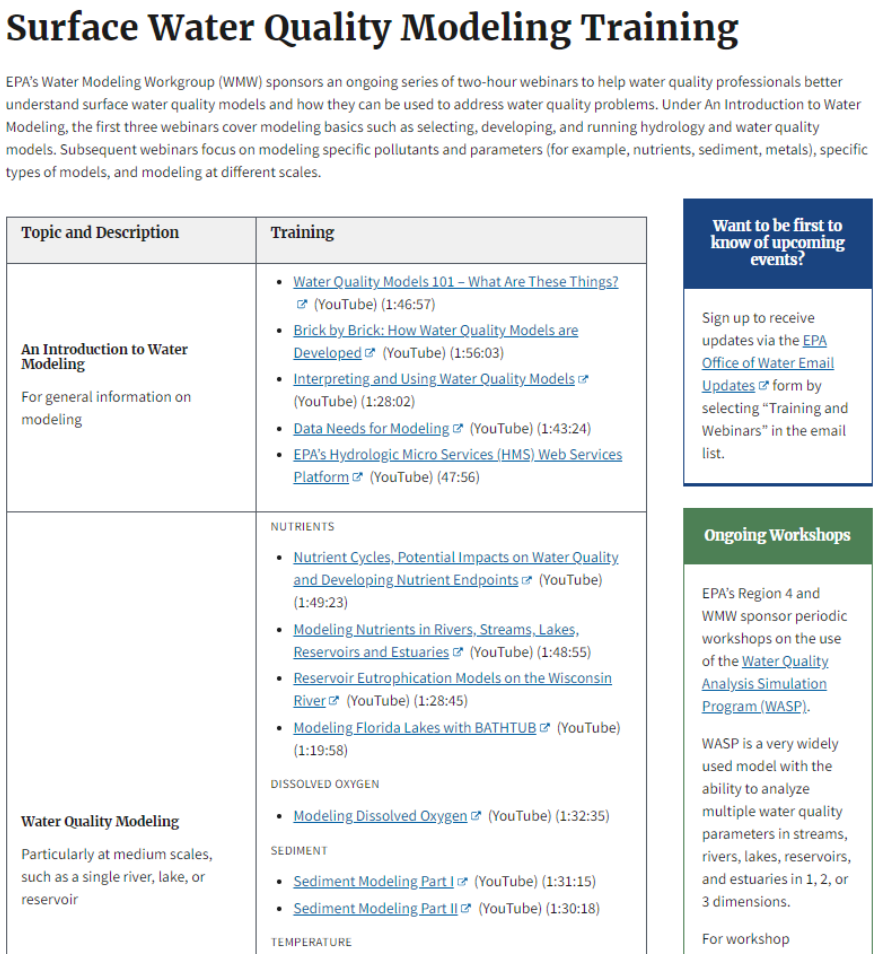
Expedia®

KAYAK

travelocity®

Example 1: Manual Extraction of Data from a Website

Manual Extraction of Data from a Website



Surface Water Quality Modeling Training

EPA's Water Modeling Workgroup (WMW) sponsors an ongoing series of two-hour webinars to help water quality professionals better understand surface water quality models and how they can be used to address water quality problems. Under An Introduction to Water Modeling, the first three webinars cover modeling basics such as selecting, developing, and running hydrology and water quality models. Subsequent webinars focus on modeling specific pollutants and parameters (for example, nutrients, sediment, metals), specific types of models, and modeling at different scales.

Topic and Description	Training
An Introduction to Water Modeling For general information on modeling	• Water Quality Models 101 – What Are These Things? (YouTube) (1:46:57) • Brick by Brick: How Water Quality Models are Developed (YouTube) (1:56:03) • Interpreting and Using Water Quality Models (YouTube) (1:28:02) • Data Needs for Modeling (YouTube) (1:43:24) • EPA's Hydrologic Micro Services (HMS) Web Services Platform (YouTube) (47:56)
Water Quality Modeling Particularly at medium scales, such as a single river, lake, or reservoir	NUTRIENTS • Nutrient Cycles, Potential Impacts on Water Quality and Developing Nutrient Endpoints (YouTube) (1:49:23) • Modeling Nutrients in Rivers, Streams, Lakes, Reservoirs and Estuaries (YouTube) (1:48:55) • Reservoir Eutrophication Models on the Wisconsin River (YouTube) (1:28:45) • Modeling Florida Lakes with BATHTUB (YouTube) (1:19:58) DISSOLVED OXYGEN • Modeling Dissolved Oxygen (YouTube) (1:32:35) SEDIMENT • Sediment Modeling Part I (YouTube) (1:31:15) • Sediment Modeling Part II (YouTube) (1:30:18) TEMPERATURE

Want to be first to know of upcoming events?

Sign up to receive updates via the EPA Office of Water Email Updates form by selecting "Training and Webinars" in the email list.

Ongoing Workshops

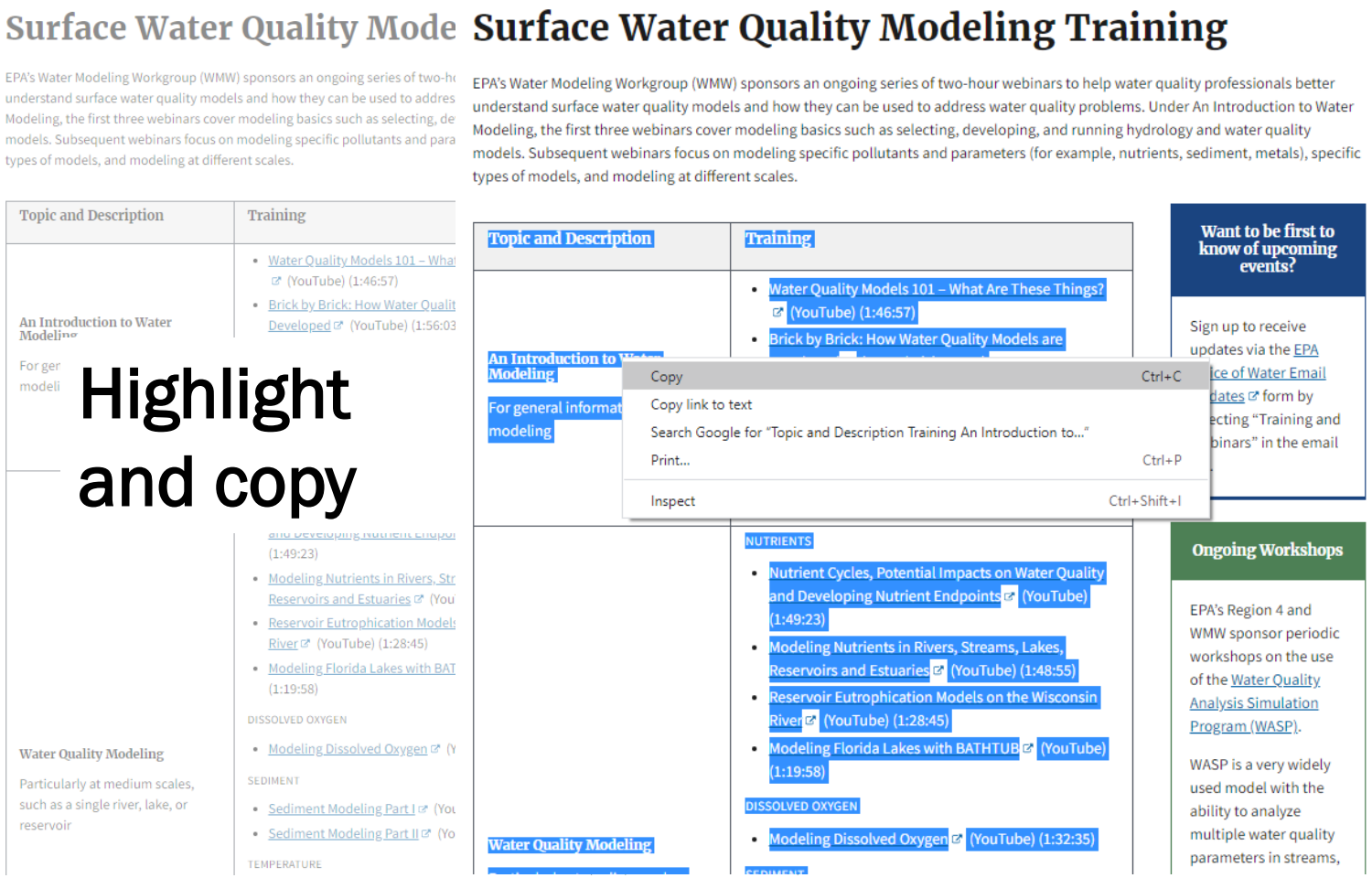
EPA's Region 4 and WMW sponsor periodic workshops on the use of the Water Quality Analysis Simulation Program (WASP).

WASP is a very widely used model with the ability to analyze multiple water quality parameters in streams, rivers, lakes, reservoirs, and estuaries in 1, 2, or 3 dimensions.

For workshop

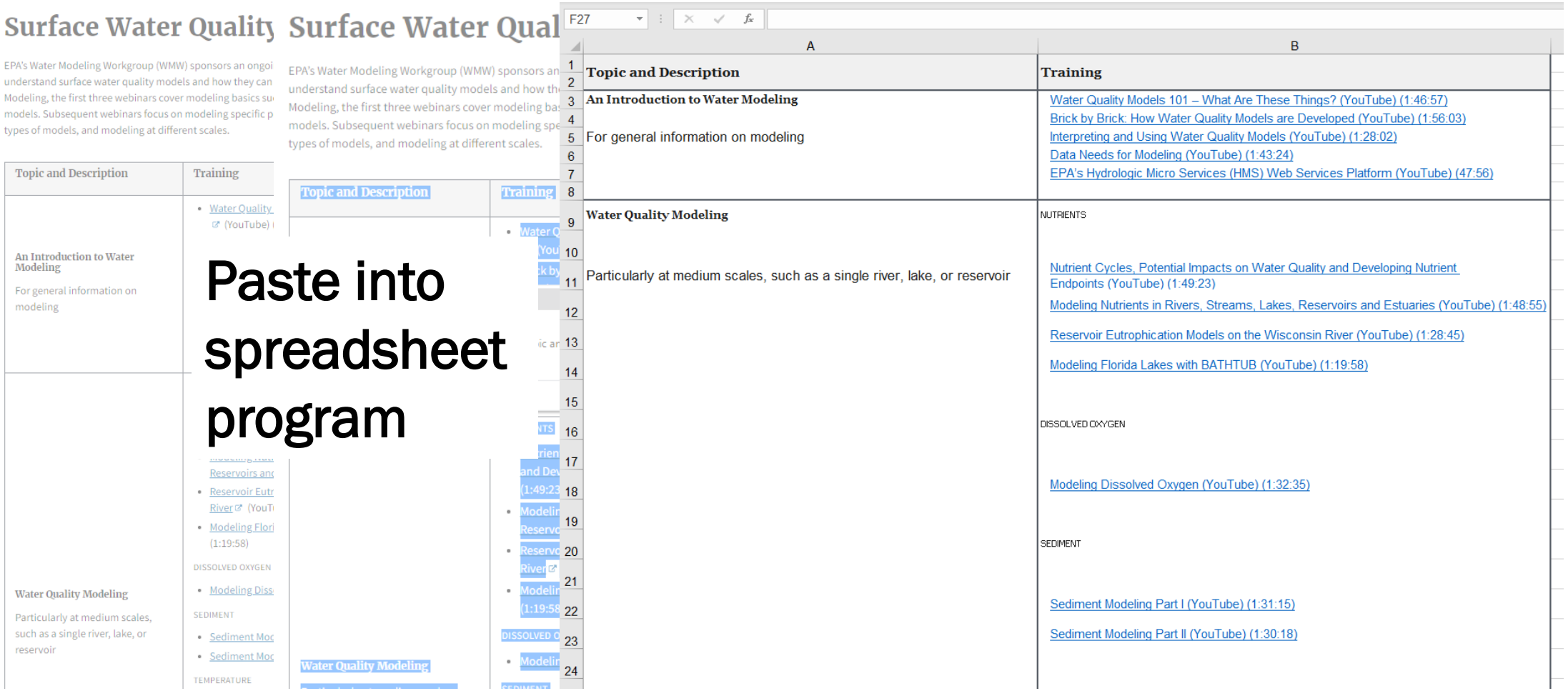
https://www.epa.gov/waterdata/surface-water-quality-modeling-training

Manual Extraction of Data from a Website



https://www.epa.gov/waterdata/surface-water-quality-modeling-training

Manual Extraction of Data from a Website



https://www.epa.gov/waterdata/surface-water-quality-modeling-training

Example 2: Automated Extraction of Data from a Website

Load HTML Code into R

Load HTML data into R using **rvest** package

```
26  library(rvest)
```

```
71  # specify the url for desired website to be scraped
72  url <- 'https://www.epa.gov/waterdata/surface-water-quality-modeling-training'
73
74  # read the HTML code from the website
75  webpage <- rvest::read_html(url)
76
```

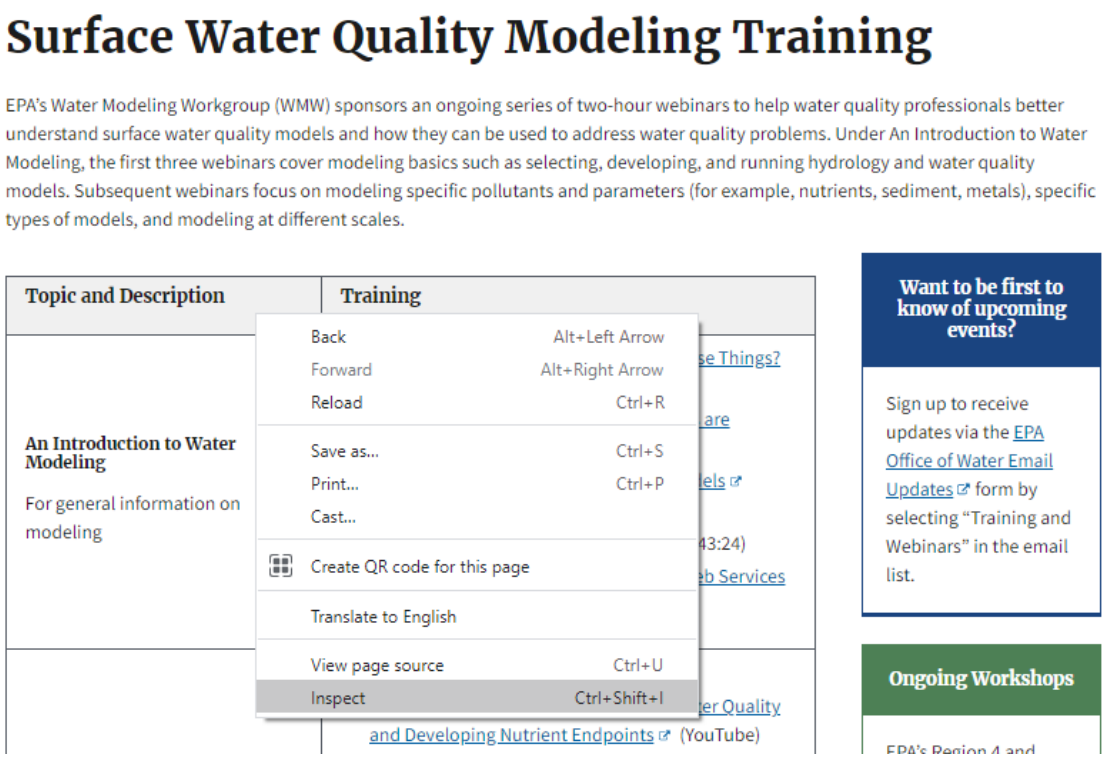
Load HTML Code into R

View the content of the object (Imported HTML code)

```
> xml2::xml_child(webpage, 2)
{html_node}
<body class="path-node not-front node-page node-page--node
-type-page" id="top">
 [1] <div class="skiplinks" role="navigation" aria-la ...
 [2] <noscript><iframe src="https://www.googletagmana ...
 [3] <div class="dialog-off-canvas-main-canvas" data- ...
 [4] <a href="#top" class="back-to-top" title="Back t ...
 [5] <script type="application/json" data-drupal-sele ...
 [6] <script src="/core/assets/vendor/jquery/jquery.m ...
```

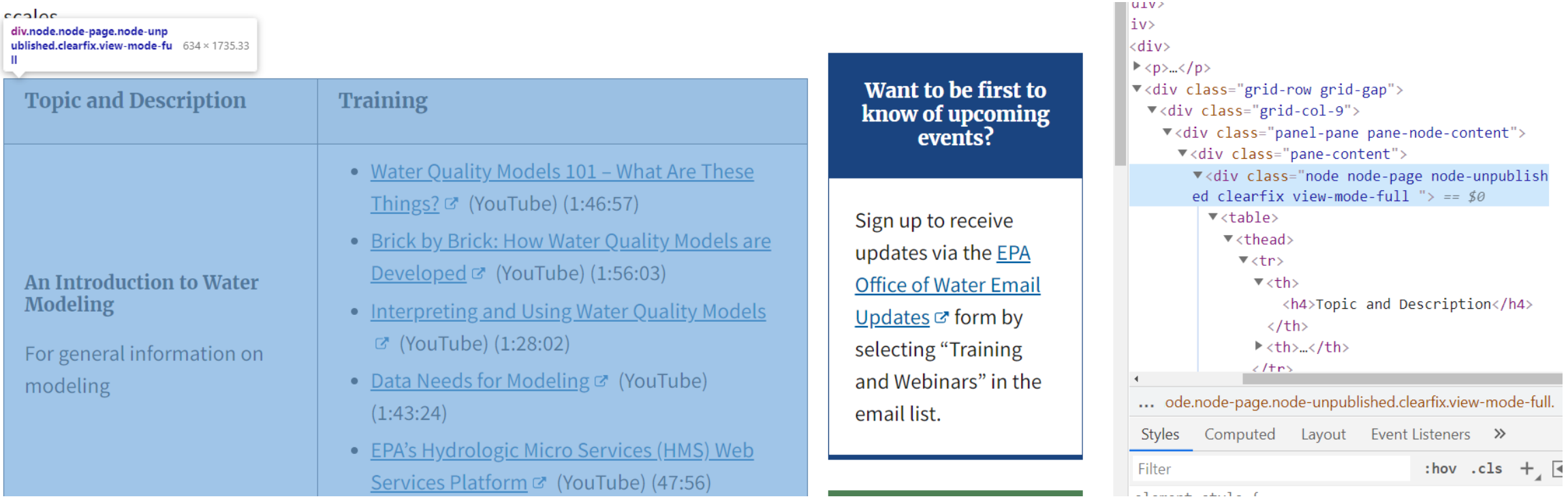
Inspect HTML Code on Website

Inspect the code on the website



Inspect HTML Code on Website

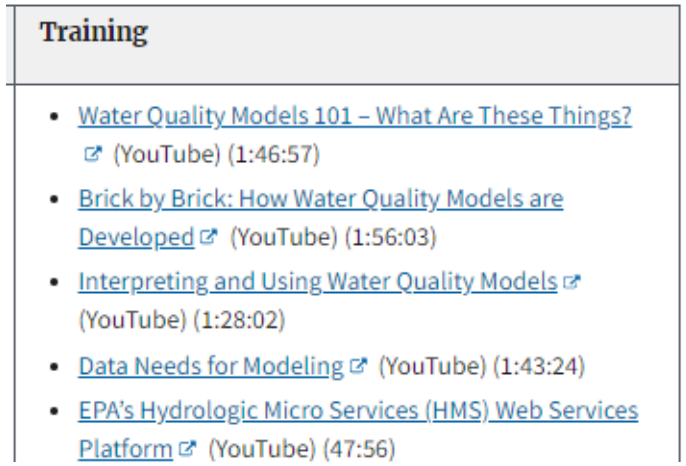
Identify the node in the HTML code



Extract Content from HTML Code

Load the appropriate node from HTML code into R

```
77  # copy the name of the training from the website
78  link_name <- rvest::html_nodes(webpage, '.view-mode-full a')
79
80  # save the text from the html code to a new variable
81  links_data <- rvest::html_text(link_name)
82
83  # copy the links to the website and save to a new variable
84  links_html <- rvest::html_nodes(webpage, '.view-mode-full a') %>%
85    rvest::html_attr('href')
```



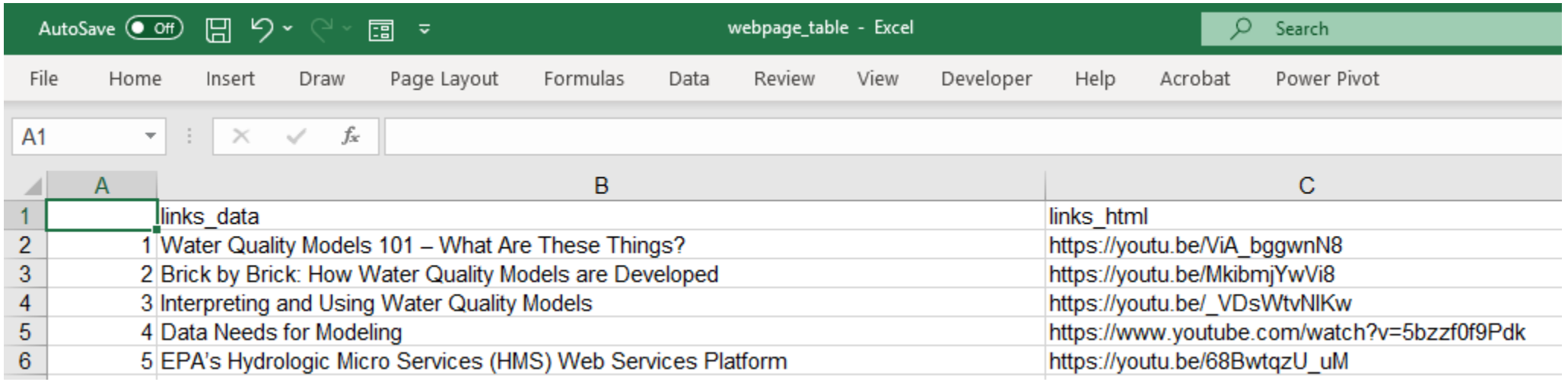

```
> links_data
 [1] "Water Quality Models 101 – What Are These Things?"

 [2] "Brick by Brick: How Water Quality Models are Develop
ed"
 [3] "Interpreting and Using Water Quality Models"

 [4] "Data Needs for Modeling"

 [5] "EPA's Hydrologic Micro Services (HMS) Web Services P
latform"
```

```
> links_html
 [1] "https://youtu.be/ViA_bggwnN8"
 [2] "https://youtu.be/MkibmjYwVi8"
 [3] "https://youtu.be/_VDsWtvNlKw"
 [4] "https://www.youtube.com/watch?v=5bzzf0f9Pdk"
 [5] "https://youtu.be/68BwtqzU_uM"
```

Extract Content from HTML Code

Combine the tables and write to a .csv file

```
87  # combine the two tables into a single table
88  webpage_table <- cbind(links_data, links_html)
89
90  # write the table into a .csv file
91  write.csv(webpage_table, file = "./Output/SWQM_webpage_table.csv")
```



	A	B	C
1		links_data	links_html
2	1	Water Quality Models 101 – What Are These Things?	https://youtu.be/ViA_bggwnN8
3	2	Brick by Brick: How Water Quality Models are Developed	https://youtu.be/MkibmjYwVi8
4	3	Interpreting and Using Water Quality Models	https://youtu.be/_VDsWtvNlKw
5	4	Data Needs for Modeling	https://www.youtube.com/watch?v=5bzzf0f9Pdk
6	5	EPA's Hydrologic Micro Services (HMS) Web Services Platform	https://youtu.be/68BwtqzU_uM

Example 3: Automated Download of Daymet Climate Data

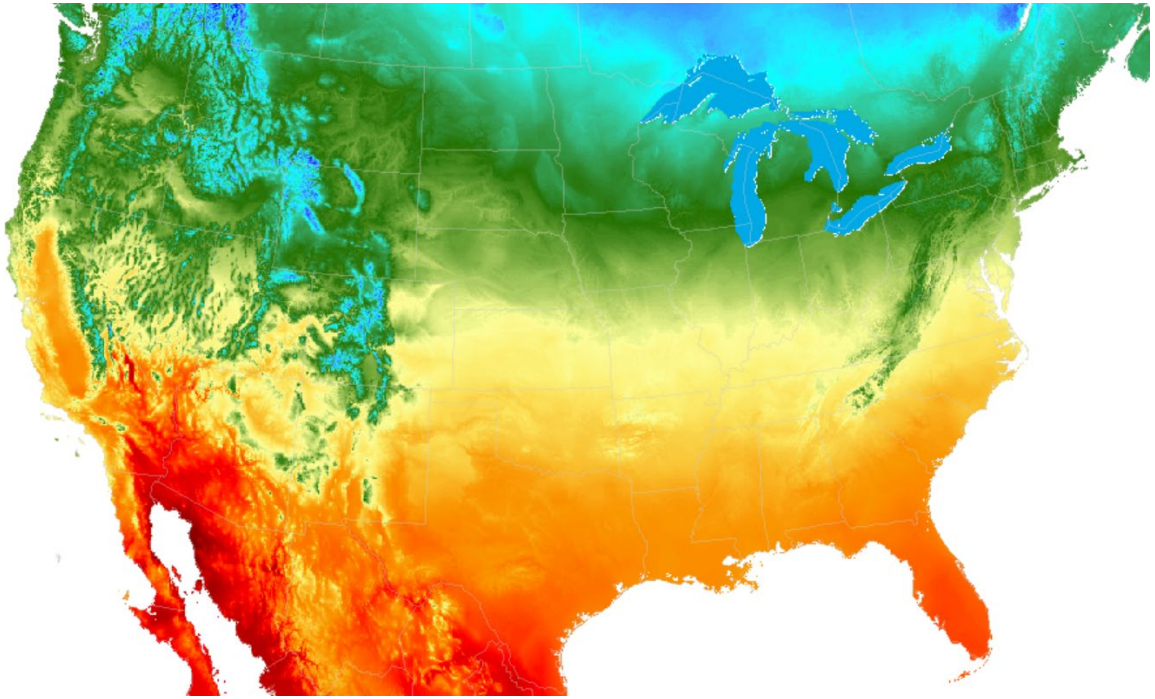
Daymet: Background

Daily Surface Weather and Climatological Summaries from Oak Ridge National Laboratory

Provides long-term, **continuous**, gridded estimates of daily weather and climate variables

Uses interpolation and extrapolation of ground-based observations to produce estimates

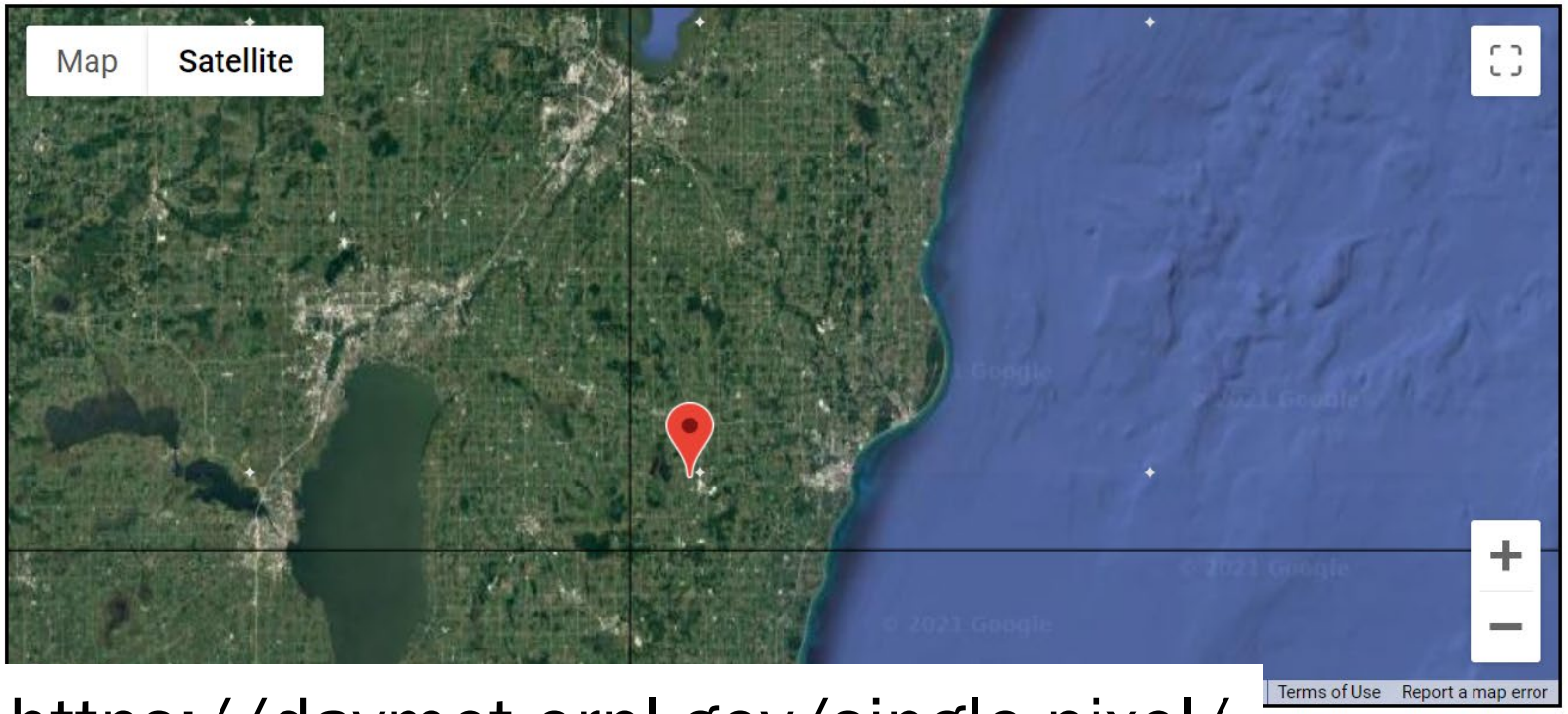
Available at 1 km x 1 km scale



Daymet: Interface

Single Pixel Extraction Tool

Coordinates for a latitude, longitude pair (in decimal degrees) can be manually entered below or automatically filled by dragging the balloon in the map. Click on a tile (within the Google Map) to see the latitude and longitude bounds for that tile.



https://daymet.ornl.gov/single-pixel/

Daymet: Interface

Single Pixel Ex

Coordinates for a latitude, longit
dragging the balloon in the map.

Coordinates:

Latitude:

44.081714

Longitude:

-87.903322

Variables:

ALL
DAYL
PRCP
SRAD
SWE
TMAX
TMIN
VP

Date Range ◉ **Years** ○

Start Date:

1990-01-01

End Date:

2020-12-31

Browser URL requests are populated as variables and time subsets are selected in the GUI. See the Web Services page for additional information.

curl -J 'https://daymet.ornl.gov/single-pixel/api/data?lat=35.9621&lon=-84.2916&vars=dayl,prcp,srad,swe,tmax,tmin,vp&start=1990-01-01&end=2006-12-31' -O

wget --content-disposition 'https://daymet.ornl.gov/single-pixel/api/data?lat=35.9621&lon=-84.2916&vars=dayl,prcp,srad,swe,tmax,tmin,vp&start=1990-01-01&end=2006-12-31'

Download Data | Visualize Data | Preview Data

12107_lat_44.081714_lon_-87.903322_2021-08-23_120823.csv

Daymet: Limitations of Interface

Challenges:

Data can only be downloaded for one location at a time

File names are difficult to discern

Solution:

Scrape data using R by manipulating download URL

Browser URL requests are populated as variables and time subsets are selected in the GUI. See the Web Services page for additional information.

```
curl -J 'https://daymet.ornl.gov/single-pixel/api/data?lat=35.9621&lon=-84.2916&vars=dayl,prcp,srad,swe,tmax,tmin,vp&start=1990-01-01&end=2006-12-31' -O
```

https://daymet.ornl.gov/single-pixel/api/data?lat=35.9621&lon=-84.2916&vars=dayl,prcp,srad,swe,tmax,tmin,vp&start=1990-01-01&end=2006-12-31

Daymet: Automated Download Link

https://daymet.ornl.gov/single-pixel/api/data?lat=35.9621&lon=-84.2916& vars=dayl,prcp,srad,swe,tmax,tmin,vp&start=1990-01-01&end=2006-12-31

https://daymet.ornl.gov/single-pixel/api/data?lat=35.9621&lon=-84.2916& 1

2 vars=dayl,prcp,srad,swe,tmax,tmin,vp&start=1990-01-01&end=2006-12-31 3

1. Location: Latitude and longitude of location of interest
2. Variables: Variables of interest for download (all selected)
3. Date range: Start and end date for period of interest

***Indicates automation of web page may be possible and straightforward!**

Daymet: Automated Download Link

0 https://daymet.ornl.gov/single-pixel/api/data?lat=35.9621&lon=-84.2916& 1

2 vars=dayl,prcp,srad,swe,tmax,tmin,vp&start=1990-01-01&end=2006-12-31 3

```
20  # set the prefix for the site to be downloaded
21  site_prefix = "https://daymet.ornl.gov/single-pixel/api/data?"
22
23  # set latitude and longitude
24  latitude = 44.081714
25  longitude = -87.903322
26
27  # set variables to be downloaded
28  variables = "dayl,prcp,srad,swe,tmax,tmin,vp"
29
30  # set beginning date and end date of interest
31  begin_date = "1998-01-01"
32  end_date = "2020-12-31"
33
34  # create link to website for downloading HUC 12 data
35  subbasin_download_link=paste(site_prefix,
36                               "lat=",latitude,"&lon=",longitude,
37                               "&vars=", variables,
38                               "&start=", begin_date,"&end=",end_date,sep="")
39
```

0. Site prefix

1. Location

2. Variables

3. Date range

Combine parameters

Daymet: Automated Download Link

https://daymet.ornl.gov/single-pixel/api/data?lat=35.9621&lon=-84.2916&
vars=dayl,prcp,srad,swe,tmax,tmin,vp&start=1990-01-01&end=2006-12-31

```
20  # set the prefix for the site to be downloaded
21  site_prefix = "https://daymet.ornl.gov/single-pixel/api/data?"
```

0. Site prefix

```
> subbasin_download_link
[1] "https://daymet.ornl.gov/single-pixel/api/data?lat=44.081714&lon=
-87.903322&vars=dayl,prcp,srad,swe,tmax,tmin,vp&start=1998-01-01&end=
2020-12-31"
```

```
29
30  # set beginning date and end date of interest
31  begin_date = "1998-01-01"
32  end_date = "2020-12-31"
33
34  # create link to website for downloading HUC 12 data
35  subbasin_download_link=paste(site_prefix,
36                               "lat=",latitude,"&lon=",longitude,
37                               "&vars=", variables,
38                               "&start=", begin_date,"&end=",end_date,sep="")
39
```

3. Date range

Combine parameters

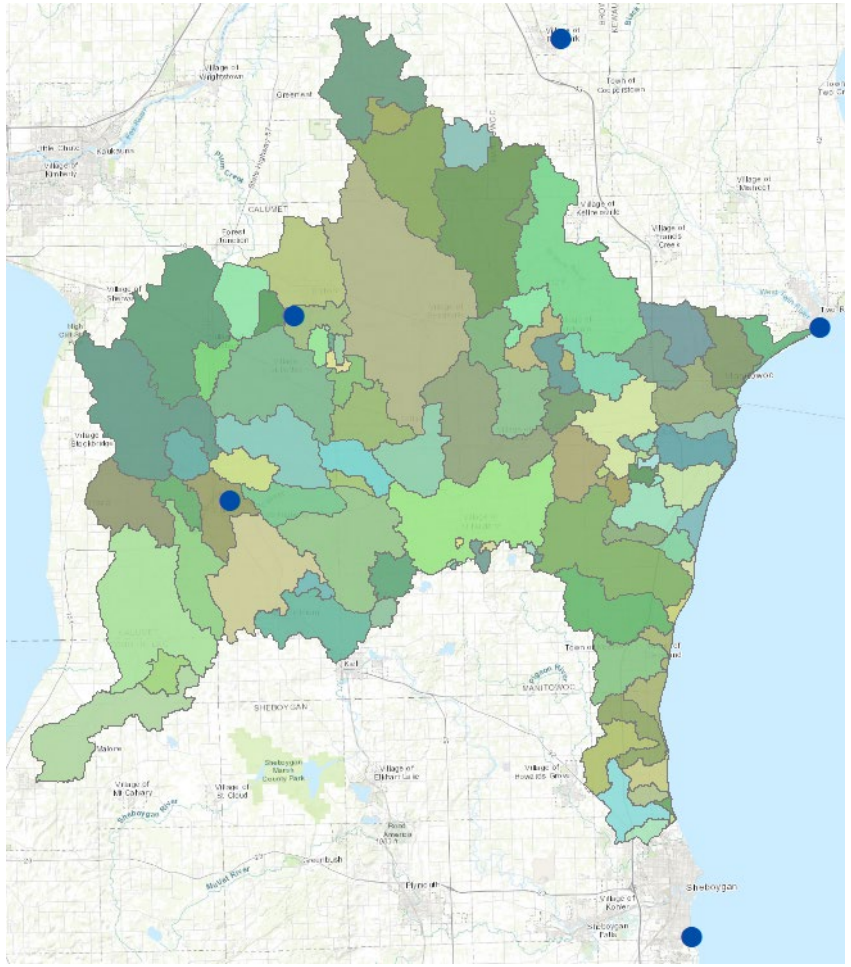
Application: Manitowoc River SWAT Model

Manitowoc River Basin (WI) SWAT Model

- 99 model subbasins
- 5 NCDC gages

Options:

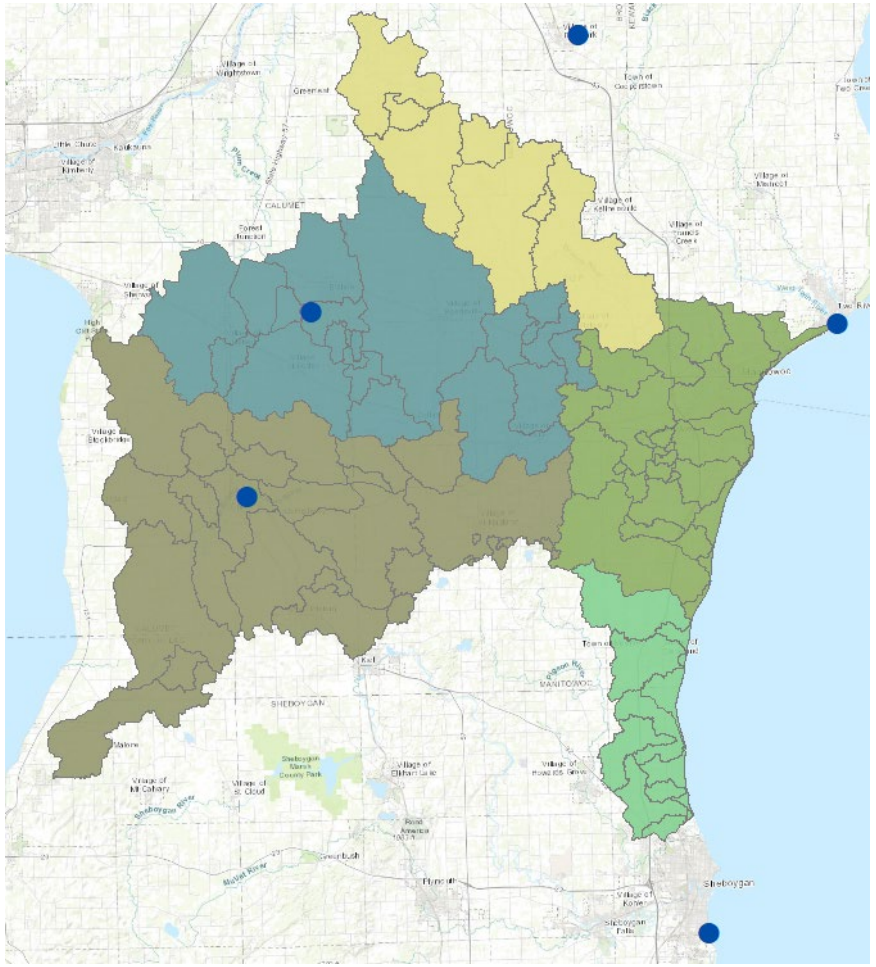
1. Use climate data from NCDC for SWAT modeling
2. Use climate data from Daymet for SWAT modeling



Manitowoc River NCDC Gages

SWAT Modeling: Climate data for each subbasin assigned to nearest gage

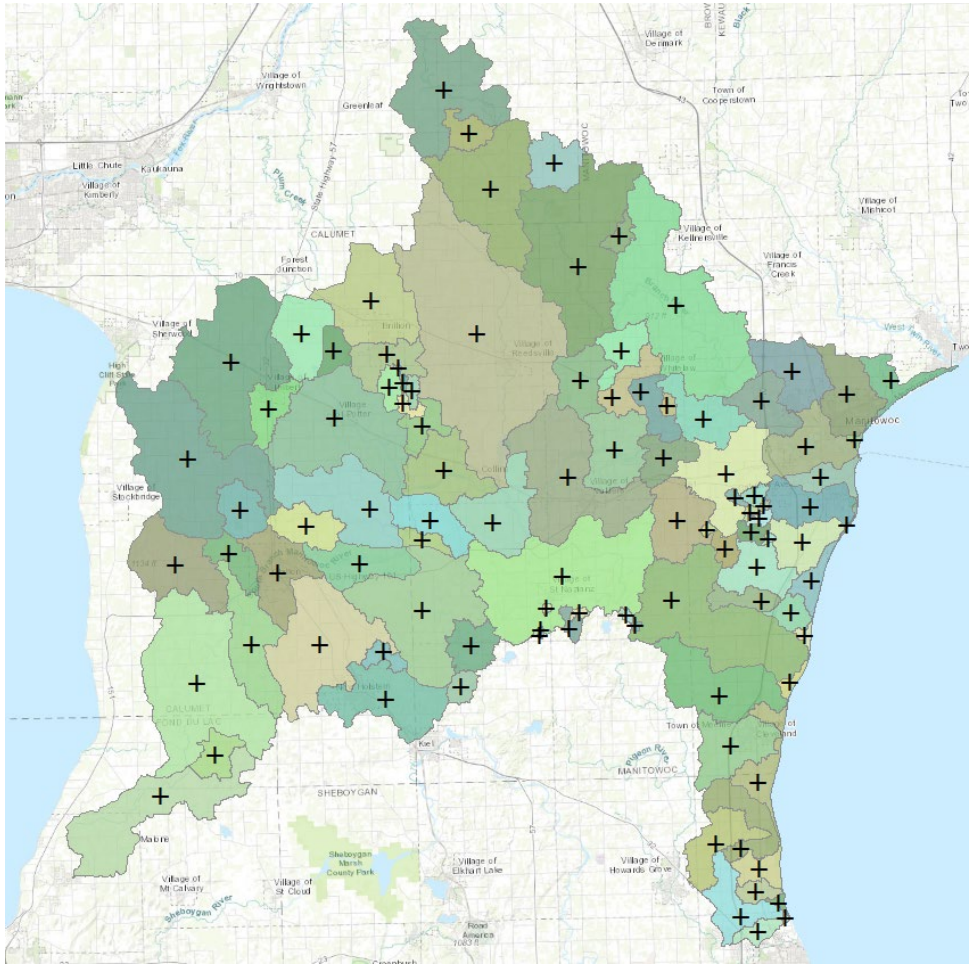
Limitation: Distance between gages may not provide accurate spatial or temporal representation of actual precipitation data



Manitowoc River SWAT Model: Daymet

Goal: Generate climate estimates for individual model subbasins

Method: Use centroid of basin for climate estimate (99 total) to extract Daymet data

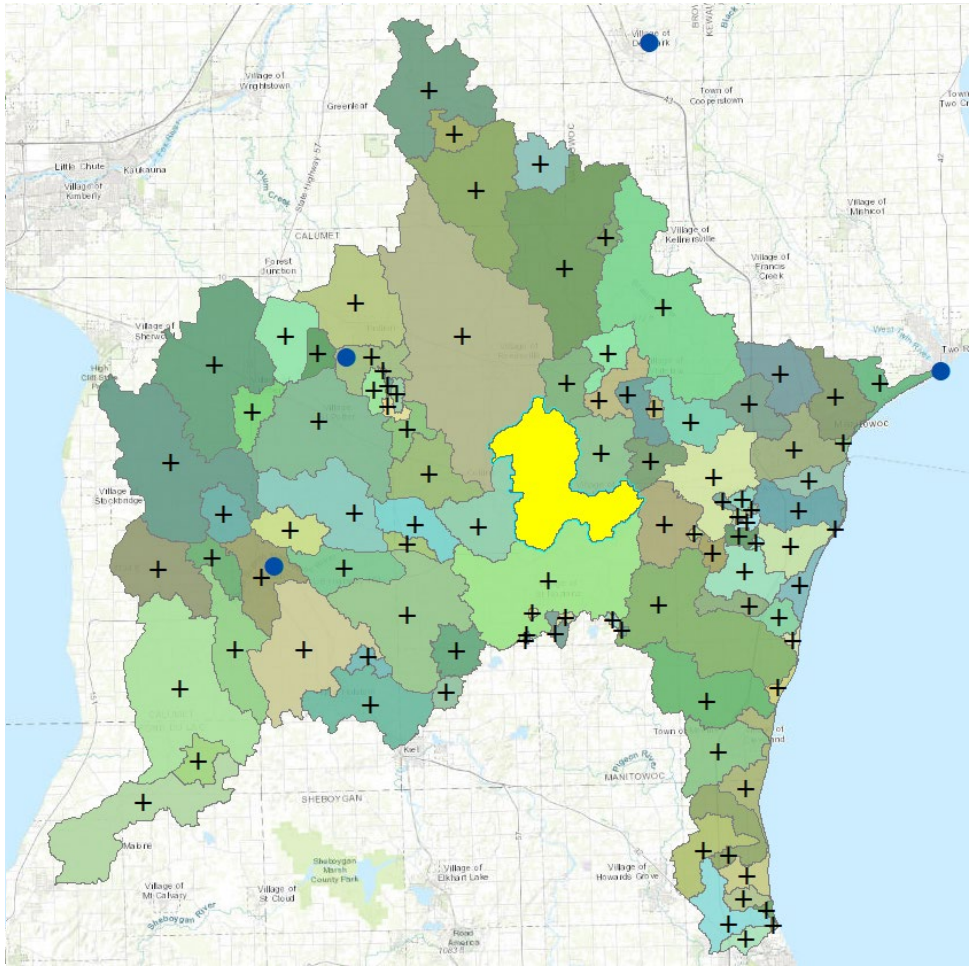


Manitowoc River SWAT Basin

Example: Subbasin at middle of basin

Between three climate stations – with SWAT and NCDC the basin would be assigned climate data from only one of those stations

Daymet uses interpolation to estimate precipitation at the specific subbasin



Download Daymet Data for Manitowoc River

Export attribute table from ArcMap to a .csv file

	A	B	C	D	E	F
1	FID	Shape *	GRIDCOD	Subbasin	Longitude	Latitude
2	0	Polygon	90	90	-87.6203	44.1369
3	1	Polygon	91	91	-87.6538	44.10037
4	2	Polygon	92	92	-87.6798	44.0773

Use read.csv function to import .csv file into R

```
113  # load information from txt file containing subbasin number, latitude, longitude
114  subbasin_data <- read.csv(file="./Input/manitowoc_subs_centroid.csv")
```

Check table in R to verify it loaded correctly

```
> subbasin_data
  FID Shape.. GRIDCODE Subbasin Longitude Latitude
1   0 Polygon       90       90 -87.62029 44.13690
2   1 Polygon       91       91 -87.65384 44.10037
3   2 Polygon       92       92 -87.67981 44.07730
4   3 Polygon       93       93 -87.69489 44.01279
```

Download Daymet Data for Manitowoc River

1. Use a loop to create a unique link for each subbasin (99 total)
2. Assign location for each subbasin
3. Create a unique file name for each subbasin
4. Generate a unique link for each subbasin
5. Use the download.file function to download the file with the specified name

```
# loop through the subbasin list to download data from Daymet
for(i in 1:nrow(subbasin_data)){

  # set subbasin, subbasin latitude, and subbasin longitude
  subbasin = subbasin_data[i,4]
  site_long = subbasin_data[i,5]
  site_lat = subbasin_data[i,6]

  # set file name for downloaded data
  file_name = paste("./Output/Daymet_downloads/manitowoc_sub_",subbasin,".csv",sep="")

  # create link to website for downloading HUC 12 data
  subbasin_download_link=paste(site_prefix,"lat=",site_lat,"&lon=",site_long,
                               "&vars=dayl,prcp,srad,swe,tmax,tmin,vp&start=",
                               begin_date,"&end=",end_date,sep="")

  # download file from daymet website
  download.file(subbasin_download_link,file_name)

} # end of for loop
```

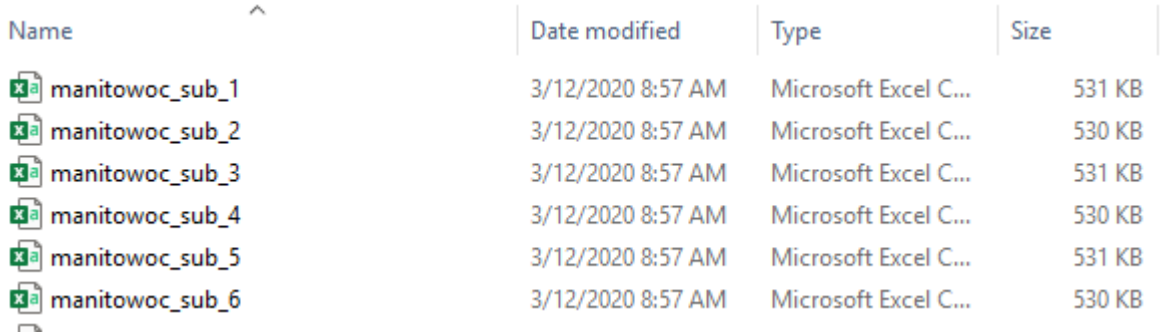
Daymet Summary: Automated Output

Summary

Final output: Collection of files with the Daymet data for the specified location and timeframe

Manual method: ~1 minute per station, 1.5 hours total

Automated method: <5 minutes total



Name	Date modified	Type	Size
manitowoc_sub_1	3/12/2020 8:57 AM	Microsoft Excel C...	531 KB
manitowoc_sub_2	3/12/2020 8:57 AM	Microsoft Excel C...	530 KB
manitowoc_sub_3	3/12/2020 8:57 AM	Microsoft Excel C...	531 KB
manitowoc_sub_4	3/12/2020 8:57 AM	Microsoft Excel C...	530 KB
manitowoc_sub_5	3/12/2020 8:57 AM	Microsoft Excel C...	531 KB
manitowoc_sub_6	3/12/2020 8:57 AM	Microsoft Excel C...	530 KB

	A	B	C	D	E	F	G	H	I
1	Latitude: 44.081714 Longitude: -87.903322								
2	X & Y on Lambert Conformal Conic: 920624.25 234422.21								
3	Tile: 12107								
4	Elevation: 257 meters								
5	All years; all variables; Daymet Software Version 3.0; Daymet Data Version 3.0.								
6	How to cite: Thornton; P.E.; M.M. Thornton; B.W. Mayer; Y. Wei; R. Devarakonda; R.S. Vose; and R.B.								
7									
8	year	yday	dayl (s)	prcp (mm/	srad (W/m	swe (kg/m	tmax (deg	tmin (deg	vp (Pa)
9	1998	1	31449.6	0	224	12	1	-10	280
10	1998	2	31449.6	0	211.2	12	6	-3.5	480
11	1998	3	31795.2	0	224	8	10.5	-1.5	560
12	1998	4	31795.2	9	163.2	8	5.5	-4.5	440
13	1998	5	31795.2	8	67.2	8	1	-2	520

Example 4: Dynamic Interaction with Websites in R Using RSelenium

Example: Manual Atlas 14 Download

NOAA ATLAS 14 POINT PRECIPITATION FREQUENCY ESTIMATES: WI

Data description

Data type: Precipitation depth | Units: English | Time series type: Partial duration

Select location

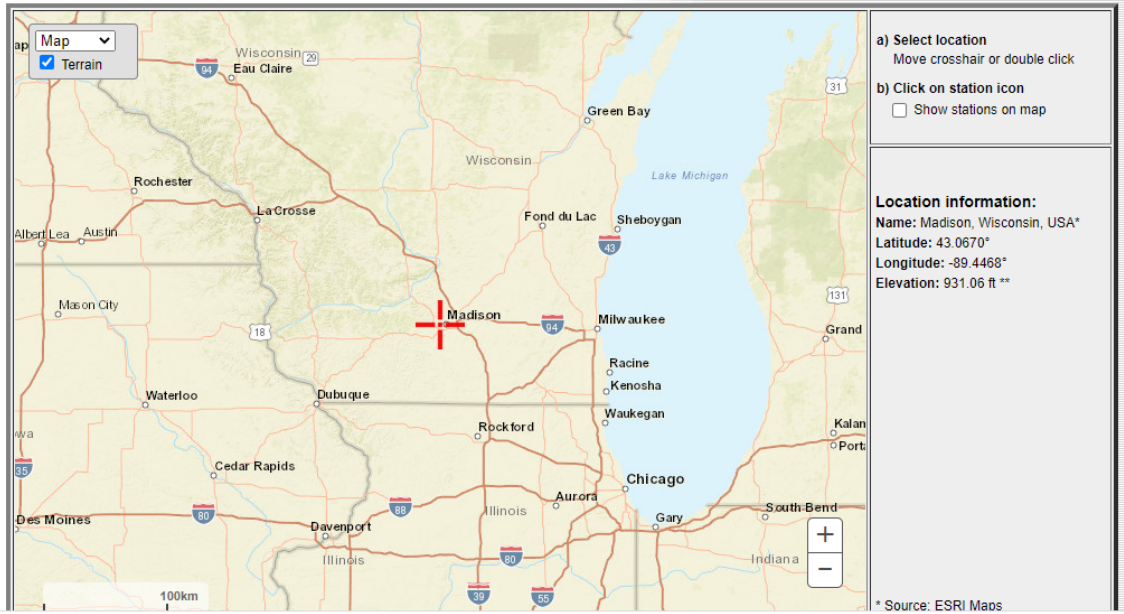
1) Manually:

a) By location (decimal degrees, use "-" for S and W): Latitude: 43.06698 Longitude: -89.44681 Submit

b) By station (list of WI stations): Select station

c) By address: Search

2) Use map (if ESRI interactive map is not loading, try adding the host: **https://js.arcgis.com/** to the firewall, or contact us at hdsc.questions@noaa.gov):



POINT PRECIPITATION FREQUENCY (PF) ESTIMATES

WITH 90% CONFIDENCE INTERVALS AND SUPPLEMENTARY INFORMATION
NOAA Atlas 14, Volume 8, Version 2

PF tabular | PF graphical | Supplementary information | Print page

PDS-based precipitation frequency estimates with 90% confidence intervals (in inches)[1]

Duration	Average recurrence interval (years)									
	1	2	5	10	25	50	100	200	500	1000
5-min	0.377 (0.327-0.438)	0.430 (0.373-0.500)	0.521 (0.451-0.607)	0.601 (0.517-0.703)	0.717 (0.600-0.865)	0.811 (0.663-0.988)	0.910 (0.719-1.13)	1.01 (0.770-1.29)	1.16 (0.847-1.50)	1.27 (0.905-1.66)
10-min	0.551 (0.479-0.641)	0.629 (0.546-0.732)	0.763 (0.660-0.889)	0.880 (0.757-1.03)	1.05 (0.879-1.27)	1.19 (0.971-1.45)	1.33 (1.05-1.65)	1.49 (1.13-1.88)	1.70 (1.24-2.20)	1.86 (1.33-2.44)
15-min	0.672 (0.584-0.781)	0.767 (0.666-0.892)	0.931 (0.805-1.08)	1.07 (0.924-1.25)	1.28 (1.07-1.55)	1.45 (1.18-1.77)	1.63 (1.28-2.02)	1.81 (1.38-2.30)	2.07 (1.51-2.68)	2.27 (1.62-2.97)
30-min	0.936 (0.813-1.09)	1.07 (0.928-1.24)	1.30 (1.12-1.51)	1.50 (1.29-1.75)	1.79 (1.50-2.16)	2.03 (1.66-2.47)	2.27 (1.80-2.82)	2.53 (1.92-3.21)	2.89 (2.11-3.75)	3.18 (2.26-4.15)
60-min	1.19 (1.03-1.38)	1.38 (1.19-1.60)	1.69 (1.47-1.97)	1.97 (1.70-2.31)	2.38 (1.99-2.87)	2.71 (2.21-3.30)	3.05 (2.41-3.79)	3.42 (2.59-4.34)	3.92 (2.87-5.08)	4.32 (3.07-5.65)
2-hr	1.44 (1.26-1.66)	1.68 (1.47-1.94)	2.09 (1.82-2.42)	2.45 (2.12-2.84)	2.97 (2.50-3.56)	3.39 (2.79-4.11)	3.83 (3.05-4.73)	4.30 (3.29-5.42)	4.95 (3.64-6.38)	5.47 (3.91-7.10)
3-hr	1.59 (1.39-1.83)	1.86 (1.63-2.14)	2.33 (2.04-2.69)	2.75 (2.38-3.18)	3.35 (2.84-4.01)	3.85 (3.18-4.65)	4.37 (3.49-5.38)	4.93 (3.78-6.19)	5.70 (4.21-7.32)	6.31 (4.54-8.17)
6-hr	1.87 (1.65-2.13)	2.17 (1.91-2.48)	2.72 (2.38-3.11)	3.21 (2.80-3.68)	3.94 (3.36-4.70)	4.55 (3.78-5.47)	5.20 (4.18-6.37)	5.90 (4.56-7.38)	6.89 (5.13-8.80)	7.69 (5.56-9.88)
12-hr	2.18 (1.94-2.48)	2.49 (2.21-2.83)	3.06 (2.70-3.48)	3.59 (3.15-4.10)	4.41 (3.80-5.25)	5.11 (4.28-6.12)	5.86 (4.76-7.16)	6.70 (5.22-8.35)	7.89 (5.92-10.0)	8.86 (6.45-11.3)
24-hr	2.49 (2.22-2.80)	2.84 (2.53-3.20)	3.48 (3.09-3.93)	4.08 (3.60-4.62)	5.00 (4.33-5.91)	5.79 (4.88-6.89)	6.64 (5.42-8.06)	7.58 (5.95-9.38)	8.92 (6.74-11.3)	10.0 (7.34-12.7)

Example: Manual Atlas 14 Download

30-day	**7.55** (6.92-8.21)	**8.43** (7.72-9.17)	**9.85** (8.99-10.7)	**11.0** (10.00-12.0)	**12.6** (11.1-14.1)	**13.8** (11.9-15.6)	**15.0** (12.5-17.3)	**16.1** (13.0-19.0)	**17.7** (13.7-21.3)	**18.8** (14.3-23.1)
45-day	**9.41** (8.66-10.2)	**10.5** (9.65-11.4)	**12.2** (11.2-13.3)	**13.6** (12.4-14.8)	**15.3** (13.5-17.0)	**16.7** (14.4-18.7)	**17.9** (15.0-20.5)	**19.1** (15.4-22.4)	**20.6** (16.0-24.7)	**21.7** (16.5-26.5)
60-day	**11.0** (10.2-11.9)	**12.3** (11.3-13.3)	**14.3** (13.1-15.5)	**15.8** (14.4-17.2)	**17.8** (15.7-19.6)	**19.2** (16.6-21.5)	**20.5** (17.2-23.4)	**21.8** (17.6-25.4)	**23.2** (18.1-27.7)	**24.2** (18.5-29.5)

[1] Precipitation frequency (PF) estimates in this table are based on frequency analysis of partial duration series (PDS).

Numbers in parenthesis are PF estimates at lower and upper bounds of the 90% confidence interval. The probability that precipitation frequency estimates (for a given duration and average recurrence interval) will be greater than the upper bound (or less than the lower bound) is 5%. Estimates at upper bounds are not checked against probable maximum precipitation (PMP) estimates and may be higher than currently valid PMP values.

Please refer to NOAA Atlas 14 document for more information.

Estimates from the table in CSV format: Precipitation frequency estimates Submit

	A	B	C	D	E	F	G	H	I	J	K
1	Point precipitation frequency estimates (inches)										
2	NOAA Atlas 14 Volume 8 Version 2										
3	Data type: Precipitation depth										
4	Time series type: Partial duration										
5	Project area: Midwestern States										
6	Location n	Wisconsir	USA								
7	Station Name: -										
8	Latitude: 43.0670°										
9	Longitude: -89.4468°										
10	Elevation (USGS): 931.06 ft										
11											
12											
13	PRECIPITATION FREQUENCY ESTIMATES										
14	by duration	1	2	5	10	25	50	100	200	500	1000
15	5-min:	0.377	0.43	0.521	0.601	0.717	0.811	0.91	1.01	1.16	1.27
16	10-min:	0.551	0.629	0.763	0.88	1.05	1.19	1.33	1.49	1.7	1.86

RSelenium: Interacting with Browsers

Summary: R Selenium allows R to interface with the Selenium Server

Selenium Server: Standalone java program that allows users to run HTML test suites in a range of browsers

Allows users to dynamically interact with webpages

RSelenium: Download Atlas 14 Data

Start with a list of locations where Atlas 14 data are needed

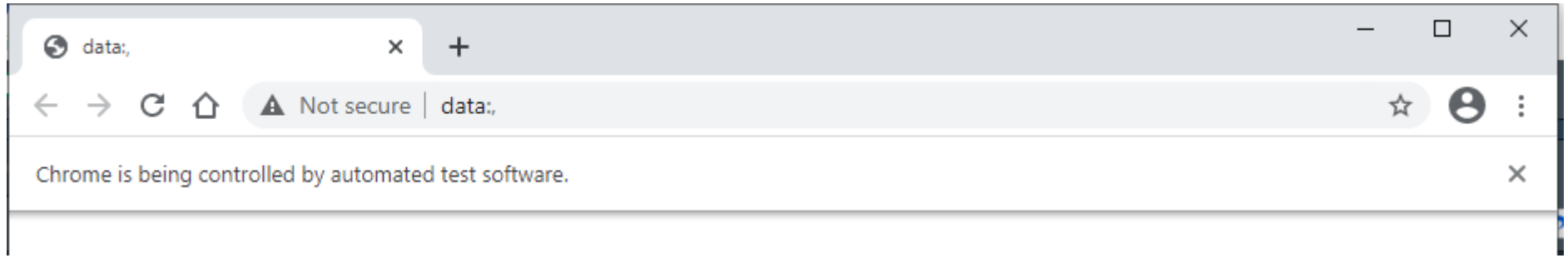
```
188  # load the atlas14_city_locations.csv data into R
189  atlas14_locations <- read.csv(file = "./Input/atlas14_city_locations.csv")
```

```
> atlas14_locations
                  city      lat       long
1 Colorado Springs, CO 43.06698  -89.44681
2     Fort Collins, CO 40.55085 -105.06681
3      Minneapolis, MN 44.97730  -93.26547
4          Lansing, MI 42.73377  -84.55538
5        Milwaukee, WI 43.03499  -87.92250
6          Madison, WI 43.07476  -89.38376
```

RSelenium: Open RSelenium Browser

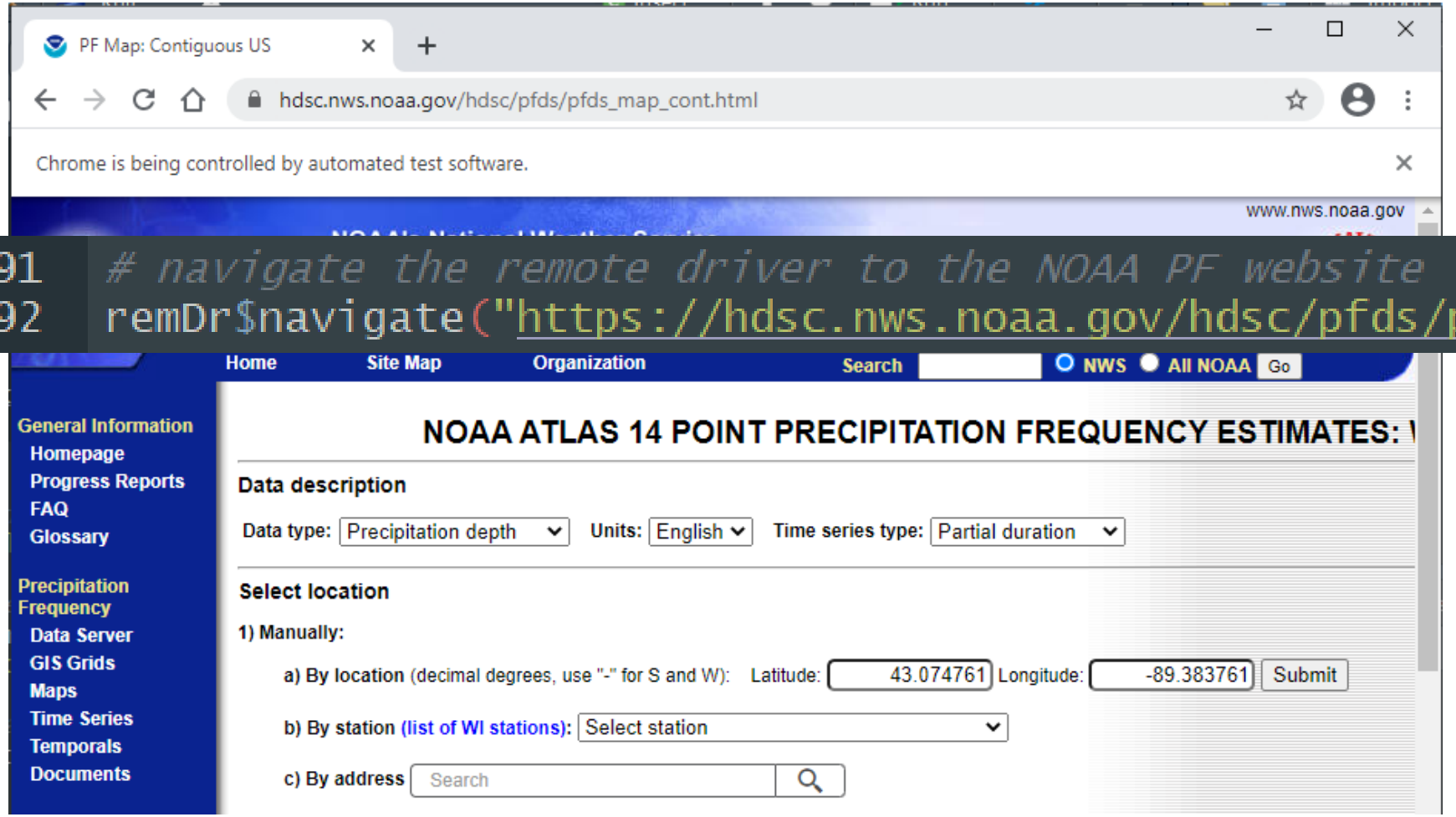
```
30  library(RSelenium)
```

```
181  # if the port is busy, run this line to clear all ports
182  system("taskkill /im java.exe /f", intern=FALSE, ignore.stdout=FALSE)
183
184  # open a remote driver from RSelenium
185  rD <- RSelenium::rsDriver(browser = "chrome", chromever = "89.0.4389.23")
186  remDr <- rD[["client"]]
```

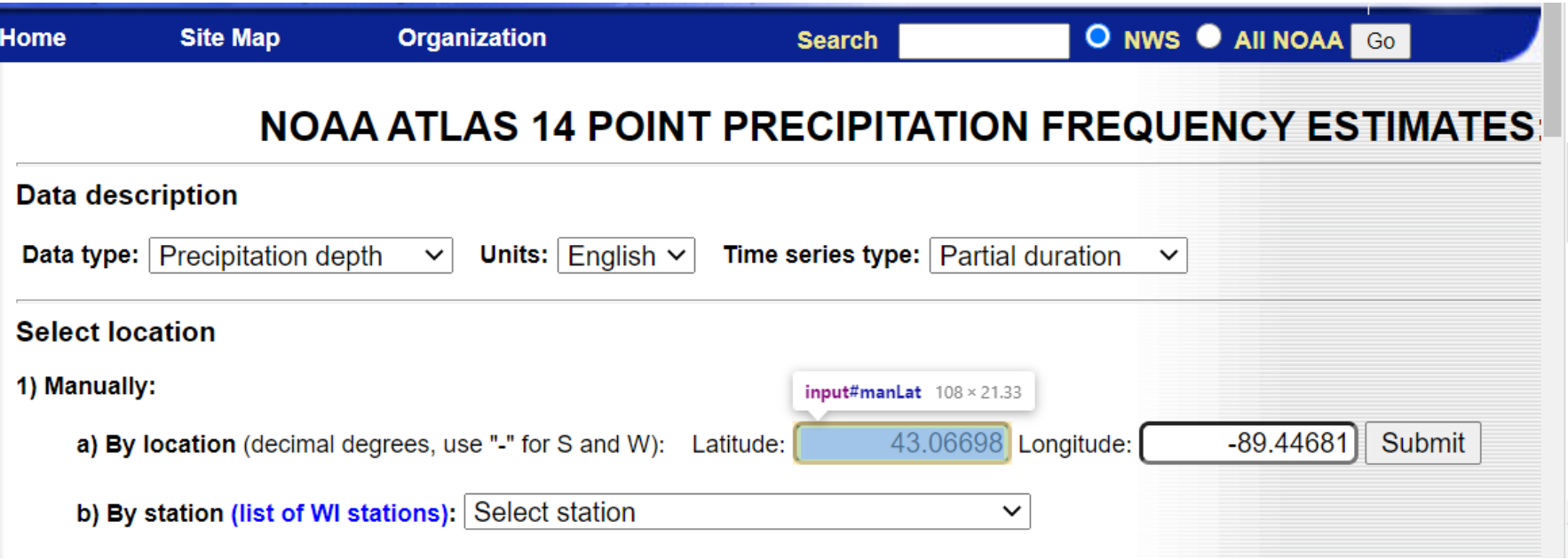


RSelenium: Navigate to site

```
191  # navigate the remote driver to the NOAA PF website
192  remDr$navigate("https://hdsc.nws.noaa.gov/hdsc/pfds/pfds_map_cont.html")
```



RSelenium: Load Latitude Value



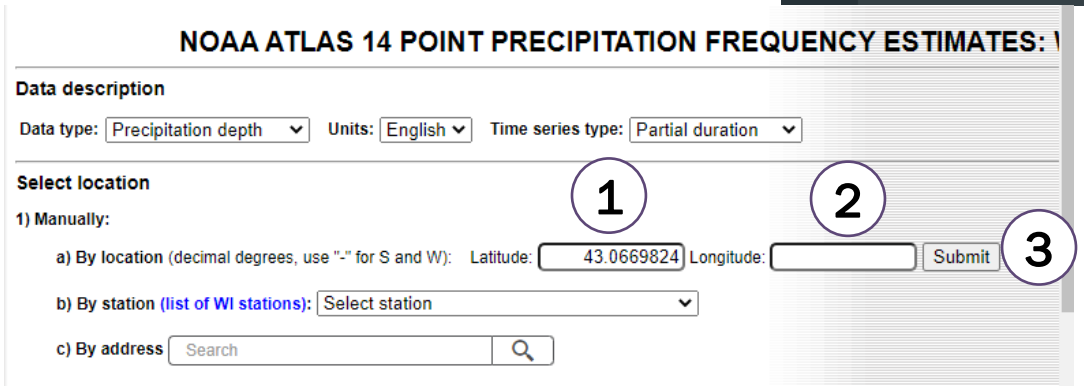

```
<span style="padding-top: 5px;">(decimal
degrees, use "-" for S and W):</span>
<label style="padding-left: 10px" for="manLat">
Latitude:</label>
<input type="text" style="width: 100px; text-ali
gn: right;" name="manLat" id="manLat" maxlength=
"10" size="14"> == $0
<label for="manLon">Longitude:</label>
<input type="text" name="manLon" id="manLon"
maxlength="10" style="width: 100px; text-align:
right;" size="14">
<input type="submit" id="latlonButton" value="Su
bmit" onclick="manualLL()">
<br>
<br>
... tbody  tr  td#rtMenu  div#locationDiv  input#manLat
```

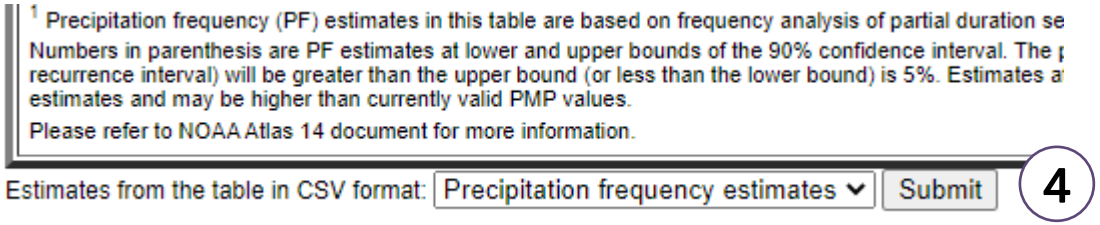
RSelenium: Automate Website Interaction

1. Enter latitude
2. Enter longitude
3. Click “Submit” button to submit lat/long data
4. Click “Submit” button to download data

```
200   # select the box on the page for inputting latitude; set latitude equal to some value
      lat_element <- remDr$findElement(using = 'id', value = "manLat")
      lat_element$clearElement()
      lat_element$sendKeysToElement(list(as.character(atlas14_locations[i,2])))
204
205   # select the box on the page for inputing longitude; set longitude equal to value
      long_element <- remDr$findElement(using = 'id', value = "manLon")
      long_element$clearElement()
      long_element$sendKeysToElement(list(as.character(atlas14_locations[i,3])))
209
210   # click the Submit button to select the location associated with the lat-long
      button_element <- remDr$findElement(using = 'id', value = "latlonButton")
      button_element$clickElement()
213
214   # pause the system to allow the data to be loaded
      Sys.sleep(1)
216
217   # identify the download button using the xpath; click the button to download
      # note: files download to "Downloads" folder
      download_button_element <- remDr$findElement(using = 'xpath', value =
'//*[@id="Table_Section"]/input')
220   download_button_element$clickElement()
```

1
2
3
4



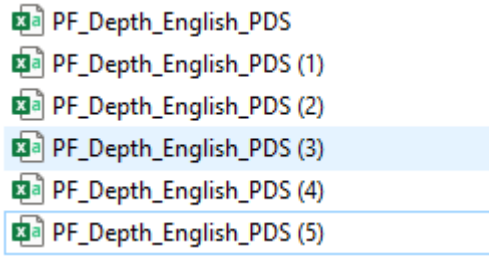


RSelenium: Automate Website Interaction

Set up loop to repeat steps for every location

```
194  # loop through the city locations to download the PFD curves
195  for(i in 1:nrow(atlas14_locations)){
196
197    #refresh the webpage
198    remDr$refresh()
199
200    # select the box on the page for inputting latitude; set latitude equal to some value
201    lat_element <- remDr$findElement(using = 'id', value = "manLat")
202    lat_element$clearElement()
203    lat_element$sendKeysToElement(list(as.character(atlas14_locations[i,2])))
204
205    # select the box on the page for inputing longitude; set longitude equal to value
206    long_element <- remDr$findElement(using = 'id', value = "manLon")
207    long_element$clearElement()
208    long_element$sendKeysToElement(list(as.character(atlas14_locations[i,3])))
209
210    # click the Submit button to select the location associated with the lat-long
211    button_element <- remDr$findElement(using = 'id', value = "latlonButton")
212    button_element$clickElement()
213
214    # pause the system to allow the data to be loaded
215    Sys.sleep(1)
216
217    # identify the download button using the xpath; click the button to download
218    # note: files download to "Downloads" folder
219    download_button_element <- remDr$findElement(using = 'xpath', value =
   '//*[@id="Table_Section"]/input')
220    download_button_element$clickElement()
221
222  }
```

Files saved to Downloads folder



CONNECT WITH US

Eric Hettler

eric.Hettler@wisconsin.gov

/WIDNR

@WIDNR

@WI_DNR

/WIDNRTV

"WILD WISCONSIN:
OFF THE RECORD"