Data is at <https://rud.is/dl/dressbarn-locations.json.gz>.

Images are in a gallery below the code.

library(rvest)

library(stringi)

library(urltools)

library(worldtilegrid) # install from sh/gl/gh or just remove the theme\_enhange\_wtg() calls

library(statebins)

library(tidyverse)

# this is the dressbarn locations directory page

pg <- read\_html("https://locations.dressbarn.com/")

# this is the selector to get the main links

html\_nodes(pg, "a.Directory-listLink") %>%

html\_attr("href") -> locs

# PRE-NOTE

# No sleep() code (I looked at the web site, saw how many self-requests it makes for all DB

# resources and concluded that link scrapes + full page captures would not be burdensome

# plus they're going out of business)

# basic idea here is to get all the main state location pages

# some states only have one store so the link goes right to that so handle that condition

# for ones with multiple stores get all the links on the state index page

# for links on state index page that have multiple stores in one area,

# grab all those; then, concatenate all the final target store links into one

# character vector.

keep(locs, ~nchar(.x) == 2) %>%

sprintf("https://locations.dressbarn.com/%s", .) %>% # state has multiple listings

map(

~read\_html(.x) %>%

html\_nodes("a.Directory-listLink") %>%

html\_attr("href") %>%

sprintf("https://locations.dressbarn.com/%s", .)

) %>%

append(

keep(locs, ~nchar(.x) > 2) %>% sprintf("https://locations.dressbarn.com/%s", .) # state has one store

) %>%

flatten\_chr() %>%

map\_if(

~stri\_count\_fixed(.x, "/") == 4, # 4 URL parts == there's another listing page layer

~read\_html(.x) %>%

html\_nodes("a.Teaser-titleLink") %>%

html\_attr("href") %>%

stri\_replace\_first\_fixed("../", "") %>%

sprintf("https://locations.dressbarn.com/%s", .)

) %>%

flatten\_chr() -> listings

# make a tibble with the HTML source for the final store location pages

# so we don't end up doing multiple retrievals

tibble(

listing = listings,

html\_src = map\_chr(listings, ~httr::GET(.x) %>% httr::content(as = "text"))

) -> dress\_barn

# save off our work in the event we have a (non-R-crashing) issue

tf <- tempfile(fileext = ".rds")

print(tf)

saveRDS(dress\_barn, tf)

# now, get data from the pages

#

# first, turn all the character vectors into something we can get HTML nodes from

#

# dressbarn web folks handliy put an "uber" link on each page so we get lon/lat for free in that URL

# they also handily used an

*semantic tag in the proper PostalAddress schema format*

*# so we can get locality and actual address, too*

*mutate(*

*dress\_barn,*

*parsed = map(html\_src, read\_html),*

*uber\_link =*

*map\_chr(*

*parsed, ~html\_nodes(.x, xpath=".//a[contains(@href, 'uber')]") %>%*

*html\_attr("href")*

*),*

*locality = map\_chr(*

*parsed, ~html\_node(.x, xpath=".//address/meta[*[*@itemprop*](http://twitter.com/itemprop) *= 'addressLocality']") %>%*

*html\_attr("content")*

*),*

*address = map\_chr(*

*parsed, ~html\_node(.x, xpath=".//address/meta[*[*@itemprop*](http://twitter.com/itemprop) *= 'streetAddress']") %>%*

*html\_attr("content")*

*),*

*state = stri\_match\_first\_regex(*

*dress\_barn$listing,*

*"https://locations.dressbarn.com/([[:alpha:]]+)/.\*$"*

*)[,2]*

*) %>%*

*bind\_cols(*

*param\_get(.$uber\_link, c("dropoff%5Blatitude%5D", "dropoff%5Blongitude%5D")) %>%*

*as\_tibble() %>%*

*set\_names(c("lat", "lon")) %>%*

*mutate\_all(as.double)*

*) -> dress\_barn*

*# save off our hard work with the HTML source so we can do more later if need be*

*select(dress\_barn, -parsed) %>%*

*saveRDS("~/Data/dressbarn-with-src.rds")*

*# save off something others will want*

*select(dress\_barn, -parsed, -html\_src, -listing) %>%*

*jsonlite::toJSON() %>%*

*write\_lines("~/Data/dressbarn-locations.json.gz")*

*# simple map*

*ggplot(dress\_barn, aes(lon, lat)) +*

*geom\_jitter(size = 0.25, color = ft\_cols$yellow, alpha = 1/2) +*

*coord\_map("polyconic") +*

*labs(*

*title = "Locations of U.S. Dressbarn Stores",*

*subtitle = "All 650 locations closing",*

*caption = "Source: Dressbarn HTML store listings;\nData: via* [*@hrbrmstr*](http://twitter.com/hrbrmstr)*"*

*) +*

*theme\_ft\_rc(grid="") +*

*theme\_enhance\_wtg()*

*unlink(tf) # cleanup*

*count(dress\_barn, state) %>%*

*left\_join(tibble(name = state.name, state = tolower(state.abb))) %>%*

*left\_join(usmap::statepop, by = c("name"="full")) %>%*

*mutate(per\_capita = (n/pop\_2015) \* 1000000) %>%*

*arrange(desc(per\_capita)) %>%*

*select(name, n, per\_capita) %>%*

*arrange(desc(per\_capita)) %>%*

*complete(name = state.name) %>%*

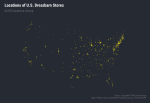
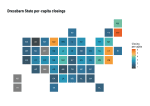
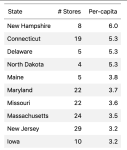
*statebins(state\_col = "name", value\_col = "per\_capita", ) +*

*scale\_fill\_r7c("Closing\nper-capita") +*

*labs(title = "Dressbarn State per-capita closings") +*

*theme\_ipsum\_rc(grid="") +*

*theme\_enhance\_wtg()*

[](https://rud.is/b/2019/05/21/add-dressbarn-to-the-continued-retailpocalypse/dressbarn-closings-map/)  
[](https://rud.is/b/2019/05/21/add-dressbarn-to-the-continued-retailpocalypse/dressbarn-per-capita-heatmap/)  
[](https://rud.is/b/2019/05/21/add-dressbarn-to-the-continued-retailpocalypse/dressbarn-top-10-per-capita/)