FOUNDATIONS OF FUNCTIONAL PROGRAMMING WITH PURRR

Using purrr in your workflow

Auriel Fournier
Instructor

Checking for names

```
library(repurrrsive)
data(sw_films)
names(sw_films)

NULL

str(sw_films)

List of 14
 $ title        : chr ...
 $ episode_id   : int ...
 $ opening_crawl: chr ...
 $ director     : chr ...
 ...
```

```
sw_films <- sw_films %>%
  set_names(map_chr(sw_films,
                    "title"))

names(sw_films)

[1] "A New Hope"
[2] "Attack of the Clones"
[3] "The Phantom Menace"
[4] "Revenge of the Sith"
[5] "Return of the Jedi"
[6] "The Empire Strikes Back"
[7] "The Force Awakens"
```

Setting names while asking questions

```
map_chr(sw_films, ~.x[["episode_id"]]) %>%
  set_names(map_chr(sw_films, "title")) %>%
  sort()
```

```
  The Phantom Menace    Attack of the Clones
                 "1"                     "2"
    Revenge of the Sith             A New Hope
                 "3"                     "4"
The Empire Strikes Back      Return of the Jedi
                 "5"                     "6"
      The Force Awakens
                 "7"
```

FOUNDATIONS OF FUNCTIONAL PROGRAMMING WITH PURRR

Let's purrr-actice!

FOUNDATIONS OF FUNCTIONAL PROGRAMMING WITH PURRR

Even more complex problems

Auriel Fournier
Instructor

What if the values you want are buried?

```
library(repurrrsive)
data(gh_repos)

forks <- gh_repos %>%
  map( ~map(.x, "forks"))

forks

[[1]]
[[1]][[1]]
[1] 0

[[1]][[2]]
[1] 1

[[1]][[3]]
[1] 0

[[1]][[4]]
[1] 0
```

Summary stats in purrr

```
# Create an empty dataframe
bird_df <- data.frame(weight=NA,
        wing_length=rep(NA, 4))

# Extract the bird names
bird_df$taxa <-
        names(bird_measurements)

# For loop to pull out each species
for(i in 1:4){
bird_df[i,] <-
        bird_measurements[[i]]
    }
summary(bird_df)
```

Summary stats

```
bird_measurements %>%
  map_df(~ data_frame(
  weight = .x[["weight"]],
  wing_length = .x[["wing length"]],
  taxa = "bird")) %>%
     select_if(is.numeric)  %>%
     summary(.x)

$weight
```

```
  Min. 1st Qu.  Median
  7.00   59.12   86.45

  Mean 3rd Qu.    Max.
  69.97   97.30  100.00

$wing_length
  Min. 1st Qu.  Median
  12.00   29.25   55.00

  Mean 3rd Qu.    Max.
  63.00   88.75  130.00
```

FOUNDATIONS OF FUNCTIONAL PROGRAMMING WITH PURRR

Let's purrr-actice!

FOUNDATIONS OF FUNCTIONAL PROGRAMMING WITH PURRR

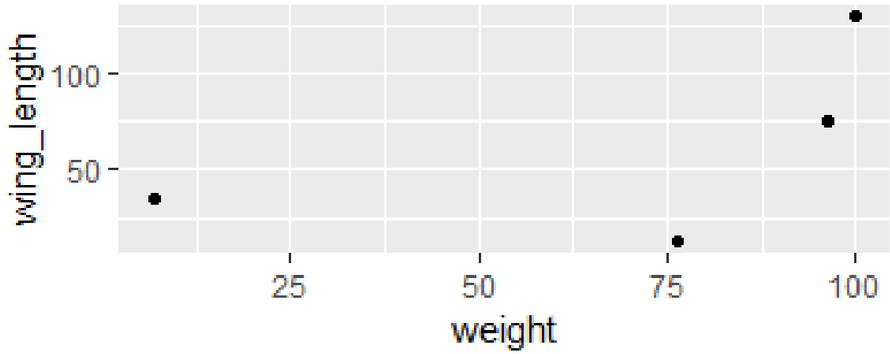
Graphs in purrr

Auriel Fournier
Instructor

ggplot() refresher

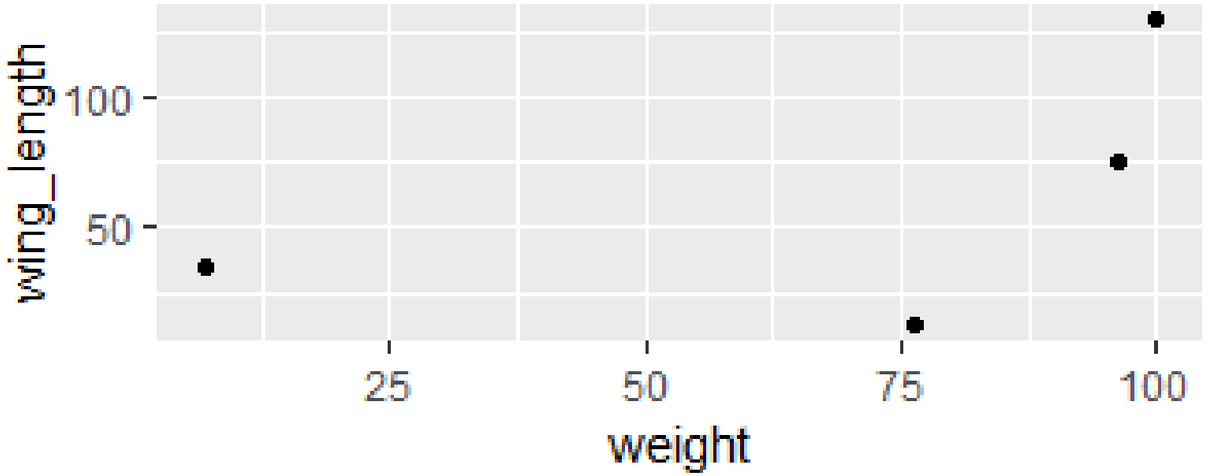
- `ggplot()` requires dataframe input
- Always start with `ggplot()`
- Add layers using +.
- `geom_*()` shows graph type

```
ggplot(data = dataframe,
       aes(x = columnA,
       y = columnB))+
  geom_point()
```



Graphing and purrr

```
birddf <- bird_measurements %>%
 map_df(~ data_frame(
            wing_length = .x[["wing length"]],
            weight = .x[["weight"]])) %>%
    ggplot(aes(x = weight,
            y = wing_length))+
    geom_point()
```



FOUNDATIONS OF FUNCTIONAL PROGRAMMING WITH PURRR

Let's purrr-actice!

FOUNDATIONS OF FUNCTIONAL PROGRAMMING WITH PURRR

Congratulations!

Auriel Fournier
Instructor

Next steps

FOUNDATIONS OF FUNCTIONAL PROGRAMMING WITH PURRR

purrrfectly done!