Working With Functionals: Takeaways

Syntax

- Applying a Function With One Variable to a List and Returning a List:

```
map(data_frame$x, function)
```

- Applying a Function with Two Variables to a List and Returning a List:

```
map2(data_frame$x, data_frame$y, function)
```

- Applying a Function with More Than Two Variables to a List and Returning a List:

```
argument_list <- list(x = data_frame$x, y = data_frame$y, z = data_frame$z)

pmap(argument_list, function)
```

USING FUNCTIONALS TO RETURN VECTORS OF SPECIFIED TYPES:

- Return a logical vector:

```
map_lgl(data_frame$x, function)

map2_lgl(data_frame$x, data_frame$y, function)

pmap_lgl(arg_list, function)
```

- Return an integer vector:

```
map_int(data_frame$x, function)

map2_int(data_frame$x, data_frame$y, function)

pmap_int(arg_list, function)
```

- Return a double vector:

```
map_dbl(data_frame$x, function)

map2_dbl(data_frame$x, data_frame$y, function)

pmap_dbl(arg_list, function)
```

- Return a character vector:

```
map_chr(data_frame$x, function)

map2_chr(data_frame$x, data_frame$y, function)

pmap_chr(arg_list, function)
```

Concepts

- Functionals take a function as an input and return a list or vector as an output. They can often be used in place of for-loops.
- The map() functionals in the purrr package return consistent output types.

Resources

- Functionals background
- purrr package documentation
- map() documentation
- map2() and pmap() documentation