Working with R Code in RStudio

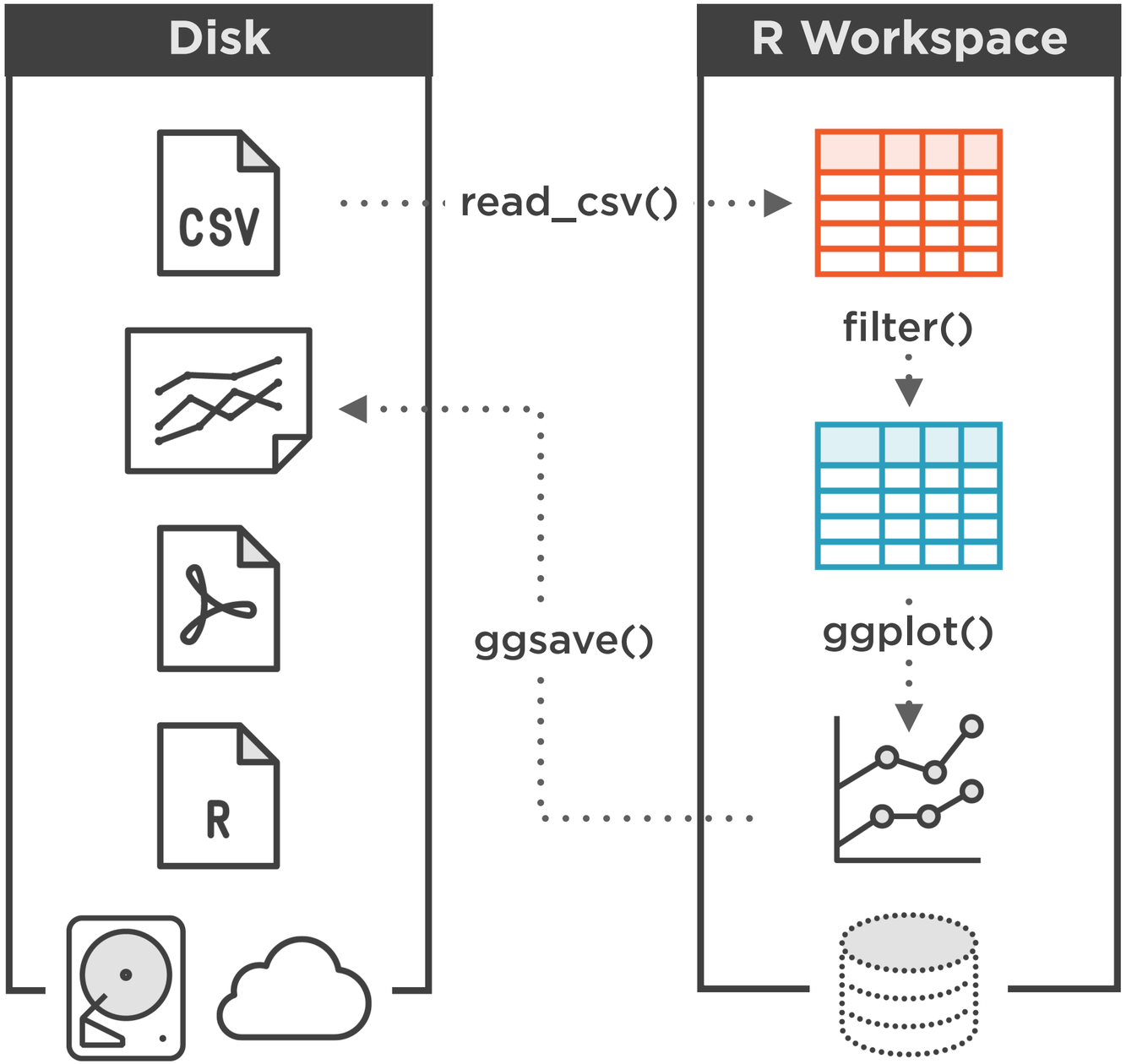
Charlotte Wickham
DATA SCIENTIST AND EDUCATOR
@cvwickham www.cwick.co.nz

Overview

Understanding the structure of R code

Writing new code

Writing reproducible scripts



Files on disk

- Save, backup and share

Objects in R workspace

- Temporary
- Instructions to create in R script

Calling R Functions

```
install.packages("tidyverse")
library(tidyverse)
```

Calling Functions

```
install.packages("tidyverse")
library(tidyverse)
```

Calling Functions

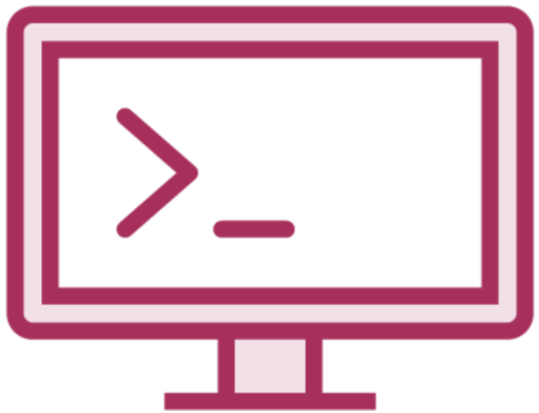
In front of parenthesis: function name

```
install.packages("tidyverse")
library(tidyverse)
```

Calling Functions

In front of parenthesis: function name
Inside parentheses: inputs

Documentation on a Function



Console

?function_name

Source Editor

Cursor in function name: F1

Passing Arguments to Functions

head(sales, n = 5)

Function Syntax

head(sales, n = 5)

Function Syntax

Function name

```
head(sales, n = 5)
```

Function Syntax

Arguments are separated by commas

head(sales, n = 5)

Function Syntax

Pass an argument without a naming it: value

```
head(sales, n = 5)
```

Function Syntax

Pass an argument by name: argument name = value

```
summarize(sales, avg_sales = mean(sales_price))
```

Understanding New R Code

```
summarize(sales, avg_sales = mean(sales_price))
```

Understanding New R Code

Call summarize with sales as first argument, and

```
summarize(sales, avg_sales = mean(sales_price))
```

Understanding New R Code

Call summarize with sales as first argument, and the avg_sales argument set to the result

```
summarize(sales, avg_sales = mean(sales_price))
```

Understanding New R Code

Call summarize with sales as first argument, and the avg_sales argument set to the result of mean called with the input sales_price.

Assigning Results to Objects

```
total_sales <- nrow(sales)
```

To Create an Object

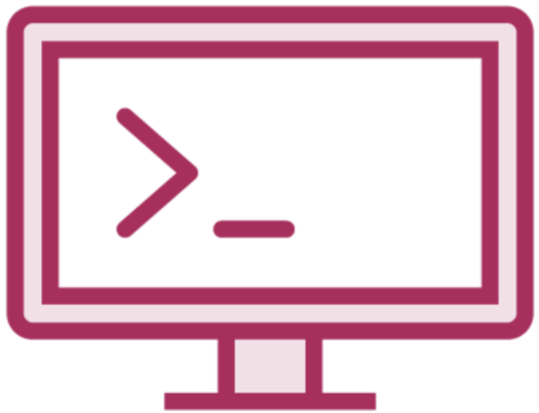
Use the assignment operator

total_sales

To Use an Object

Refer to it by its name, case sensitive

Examining an Object



Console

Type the object name

Environment

Click on object name

Understanding the Structure of an R Script

summarize_sales.R

```
library(tidyverse)

# generate some fake data that looks like the real data
sales <- data.frame(...)

# first five rows of fake data
head(sales, n = 5)

# number of rows of fake data
total_sales <- nrow(sales)
total_sales

# keep just "Dwelling" sales
dwelling_sales <- filter(sales, property_class == "Dwelling")
```

summarize_sales.R

```
library(tidyverse)

# generate some fake data that looks like the real data
sales <- data.frame(...)

# first five rows of fake data
head(sales, n = 5)

# number of rows of fake data
total_sales <- nrow(sales)
total_sales

# keep just "Dwelling" sales
dwelling_sales <- filter(sales, property_class == "Dwelling")
```

summarize_sales.R

```
library(tidyverse)

# generate some fake data that looks like the real data
sales <- data.frame(...)

# first five rows of fake data
head(sales, n = 5)

# number of rows of fake data
total_sales <- nrow(sales)
total_sales

# keep just "Dwelling" sales
dwelling_sales <- filter(sales, property_class == "Dwelling")
```

summarize_sales.R

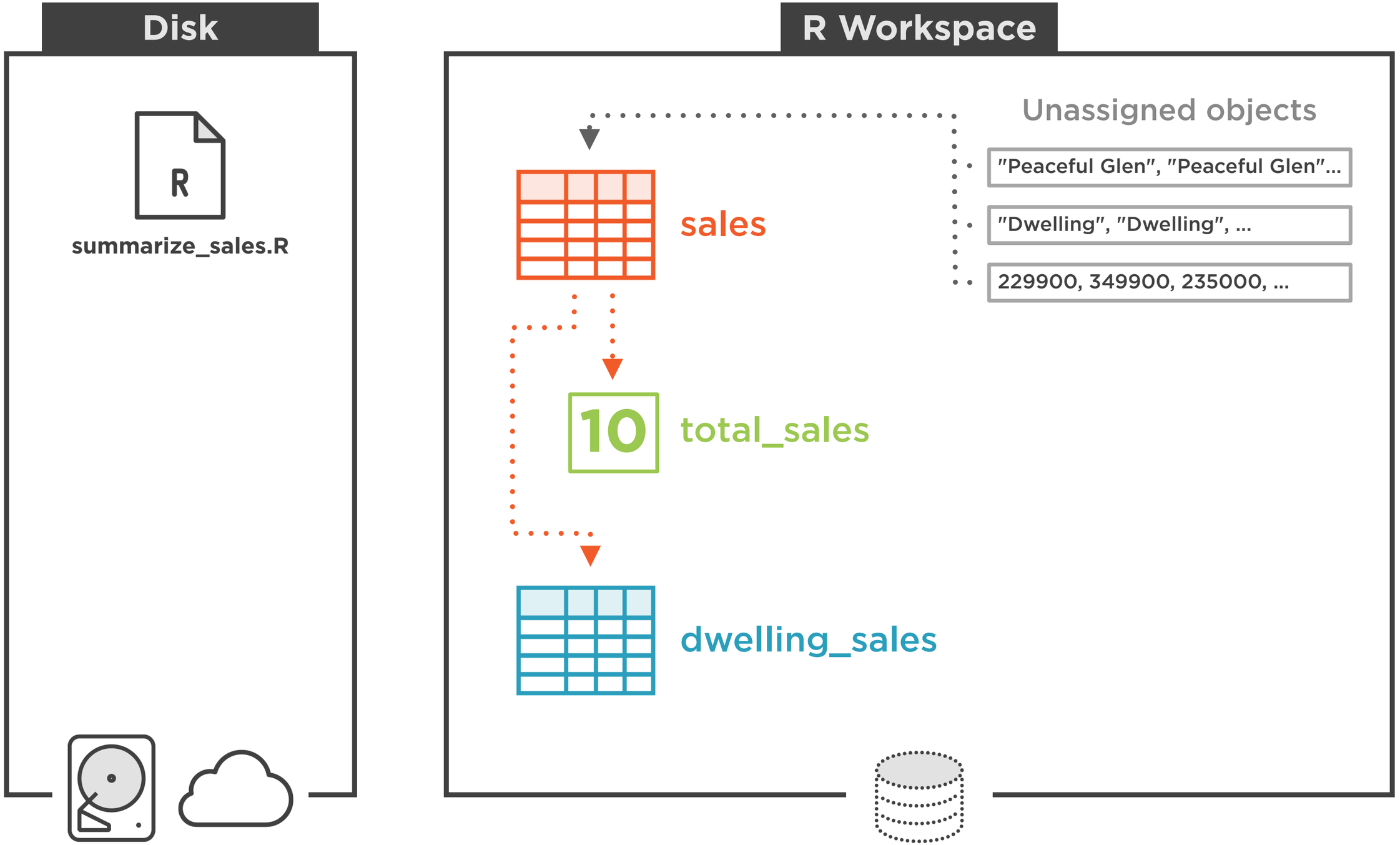
```
library(tidyverse)

# generate some fake data that looks like the real data
sales <- data.frame(...)

# first five rows of fake data
head(sales, n = 5)

# number of rows of fake data
total_sales <- nrow(sales)
total_sales

# keep just "Dwelling" sales
dwelling_sales <- filter(sales, property_class == "Dwelling")
```


Disk
R
summarize_sales.R
R Workspace
Unassigned objects
"Peaceful Glen", "Peaceful Glen"...
"Dwelling", "Dwelling", ...
229900, 349900, 235000, ...
sales
10
total_sales
dwelling_sales

summarize_sales.R

```
...

# average price by neighborhood
dwelling_sales %>%
  group_by(neighborhood) %>%
  summarize(
    n_sales = n(),
    mean_price = mean(sales_price),
    median_price = median(sales_price)
  ) %>%
  arrange(median_price)
```

summarize_sales.R

```
...

# average price by neighborhood
dwelling_sales %>%
  group_by(neighborhood) %>%
  summarize(
    n_sales = n(),
    mean_price = mean(sales_price),
    median_price = median(sales_price)
  ) %>%
  arrange(median_price)
```

```
dwelling_sales %>%
  group_by(neighborhood)
```

The Pipe Operator

Take value on the left hand side and
give as first argument to the function on the right hand side.

```
dwelling_sales %>%
  group_by(neighborhood)

group_by(dwelling_sales, neighborhood)
```

The Pipe Operator

Take value on the left hand side and
give as first argument to the function on the right hand side.

summarize_sales.R

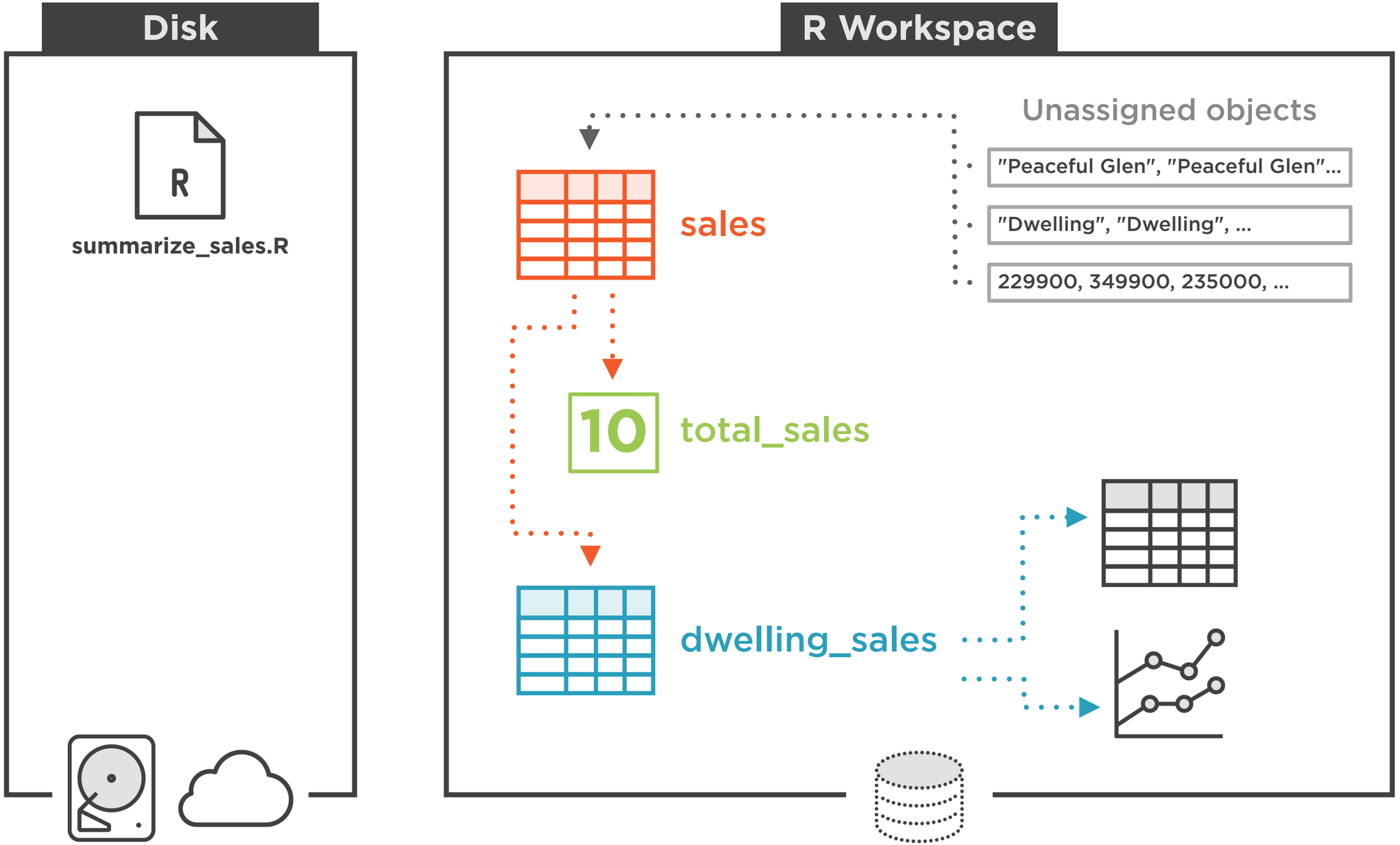
```
...

# average price by neighborhood
dwelling_sales %>%
  group_by(neighborhood) %>%
  summarize(
    n_sales = n(),
    mean_price = mean(sales_price),
    median_price = median(sales_price)
  ) %>%
  arrange(median_price)
```

summarize_sales.R

```
...

# plot all sale prices by neighborhood
dwelling_sales %>%
  ggplot(aes(sales_price)) +
  geom_histogram() +
  scale_x_log10(labels = scales::label_dollar()) +
  facet_wrap(~ neighborhood, scale = "free_y") +
  labs(
    x = "Sales price",
    y = "Number of sales"
  )
```


Disk
R
summarize_sales.R
R Workspace
Unassigned objects
"Peaceful Glen", "Peaceful Glen"...
"Dwelling", "Dwelling", ...
229900, 349900, 235000, ...
sales
10
total_sales
dwelling_sales

Writing R Code

Writing Reproducible R Scripts

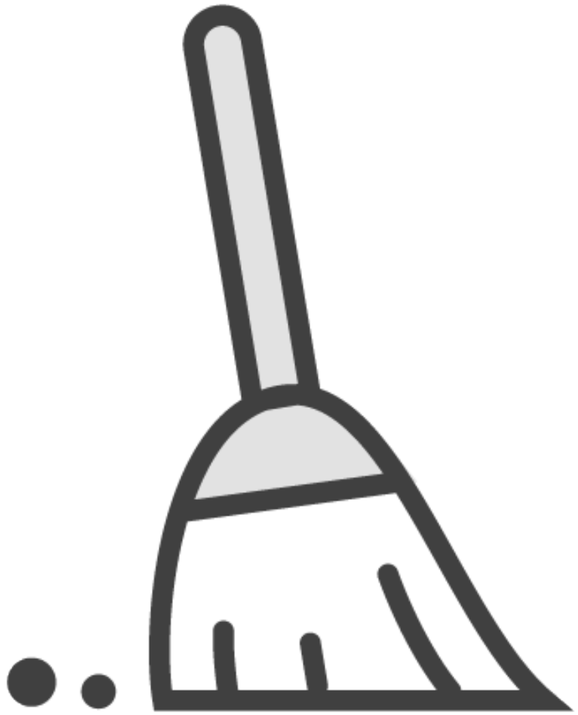
Goal: Reproducible R Scripts

Scripts should be complete and self-contained

You can reproduce your work, and others can reproduce your work

Check by restarting R and rerunning your script

Starting with a Fresh R Session

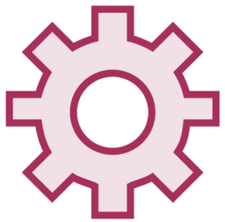


No code has been run

No packages loaded

No objects in the workspace

Ensuring Reproducible Scripts



Set RStudio to not save or load workspace

Restart R often

Source your script and check for errors

Summary

R syntax

- Calling functions
- Assigning objects

Accessing function documentation

Examining objects

Workflow

- Writing new code
- Ensuring reproducible scripts

Up next:
Organizing Your Work with RStudio Projects