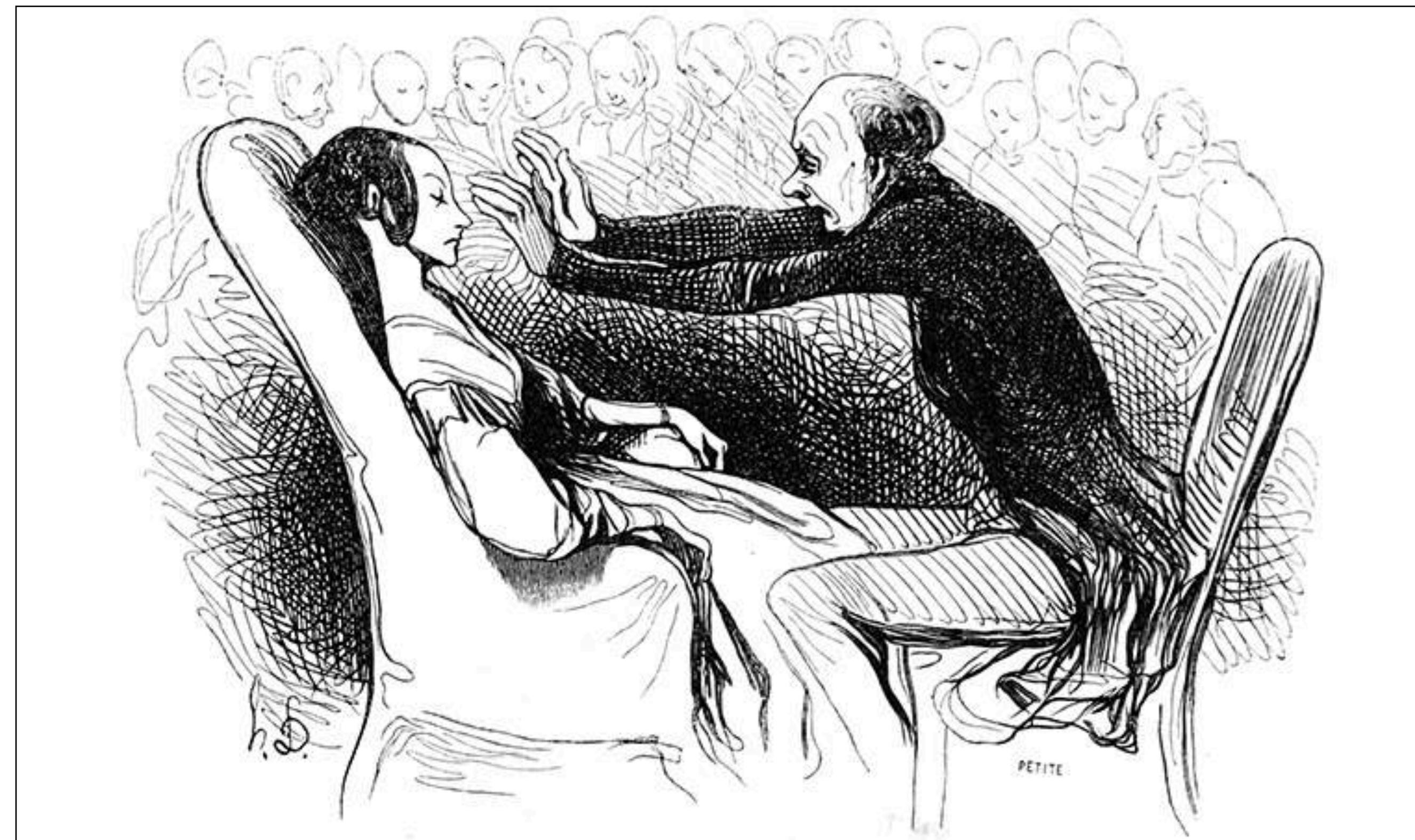


Make it Stick



The brain revisited

R

make it stick

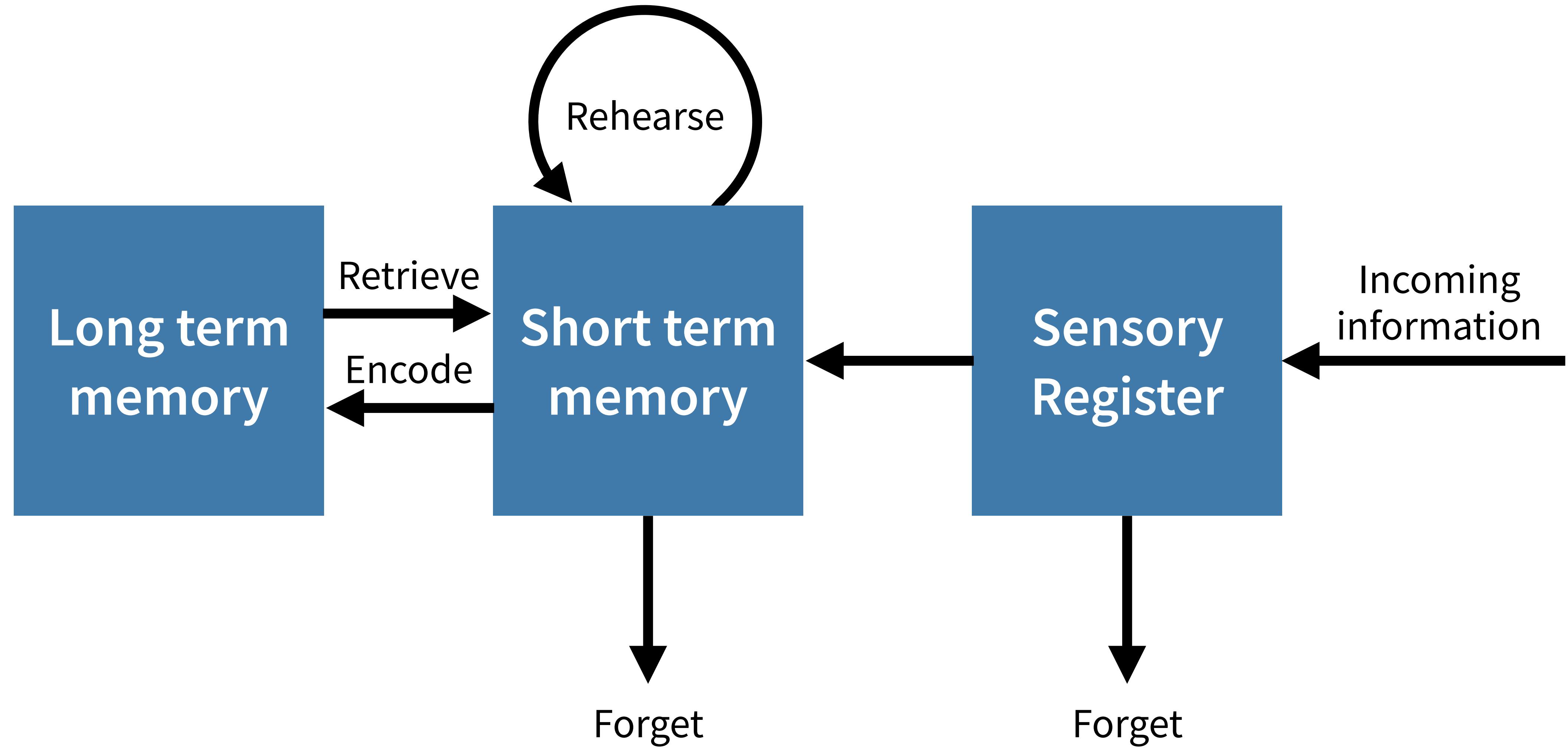


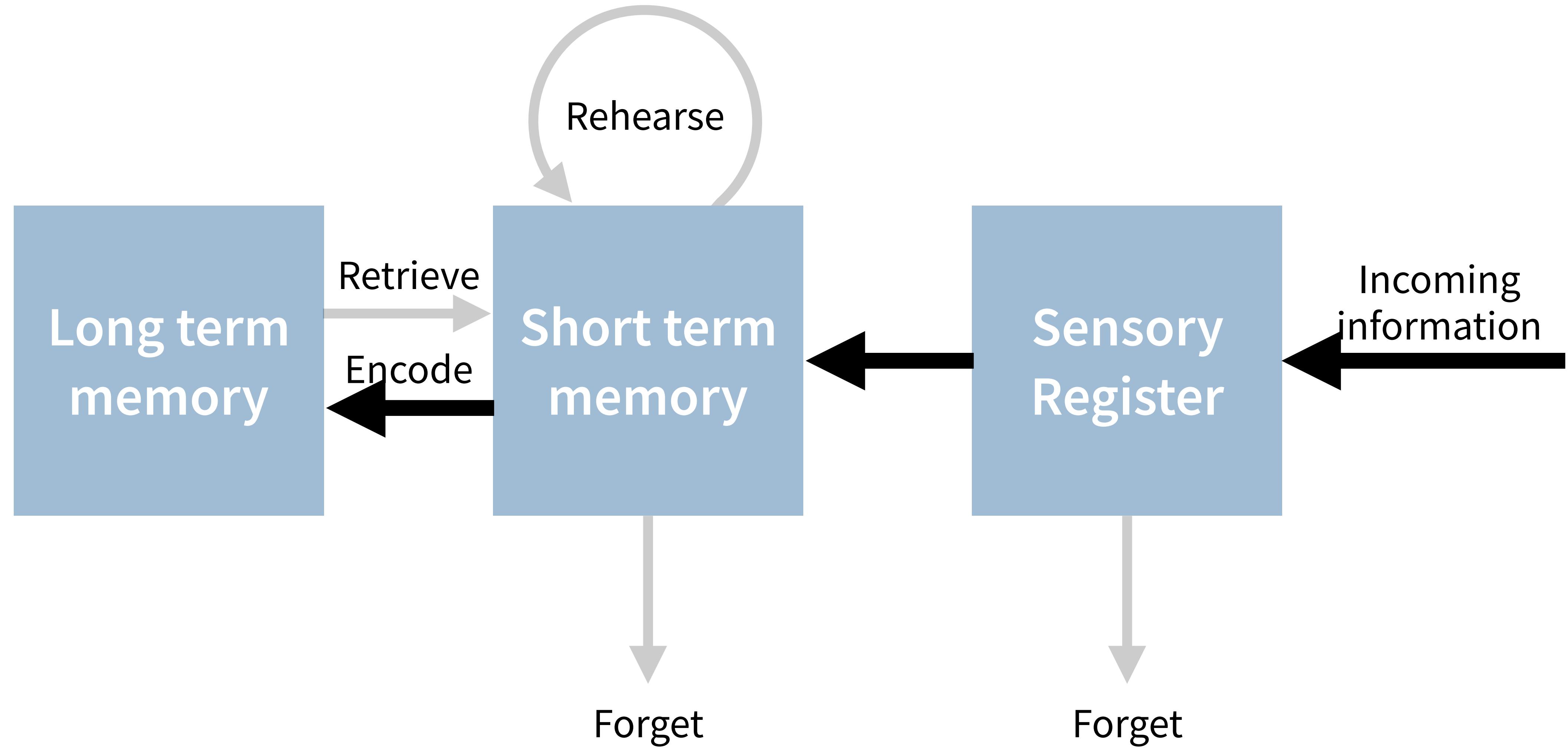
The Science of Successful Learning

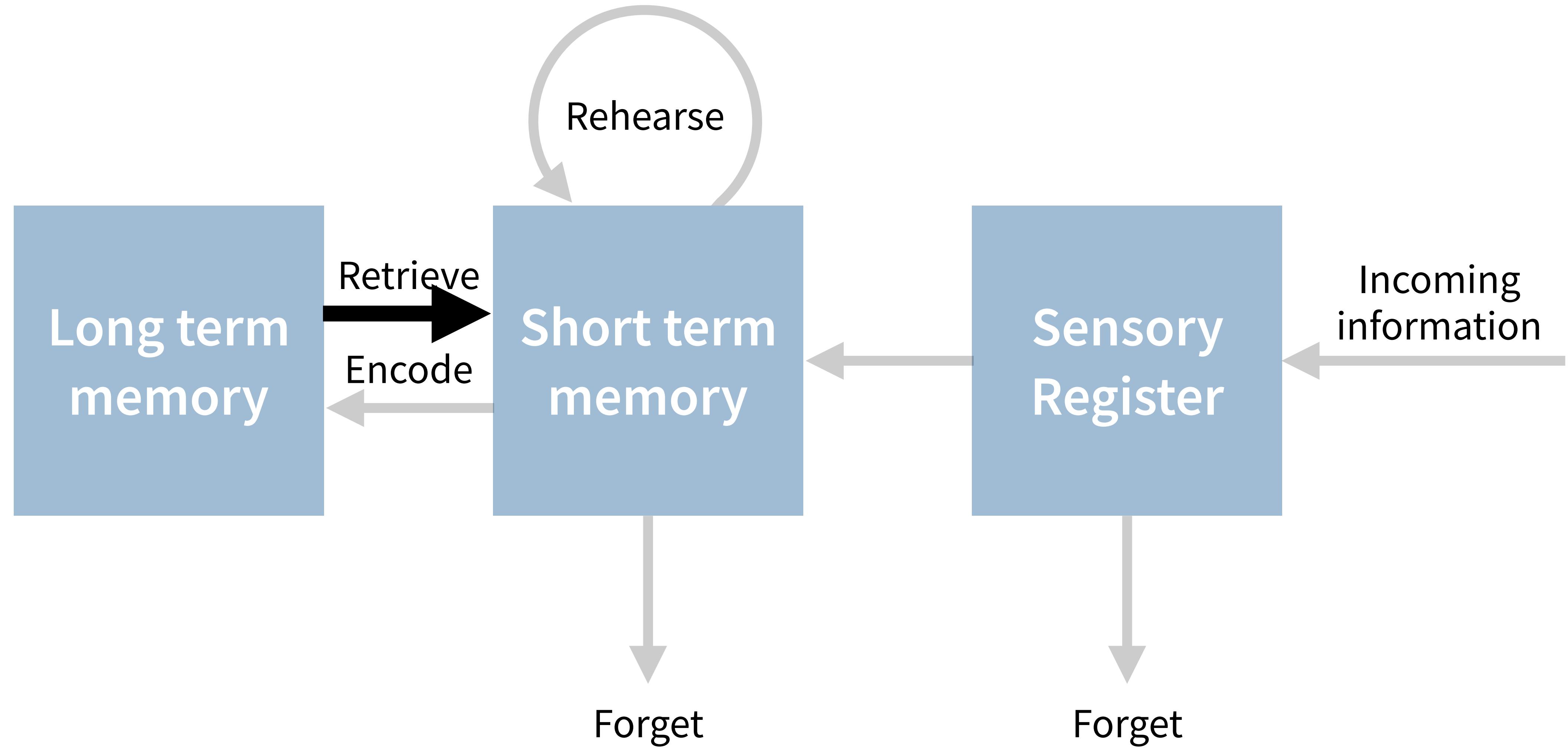
Peter C. Brown

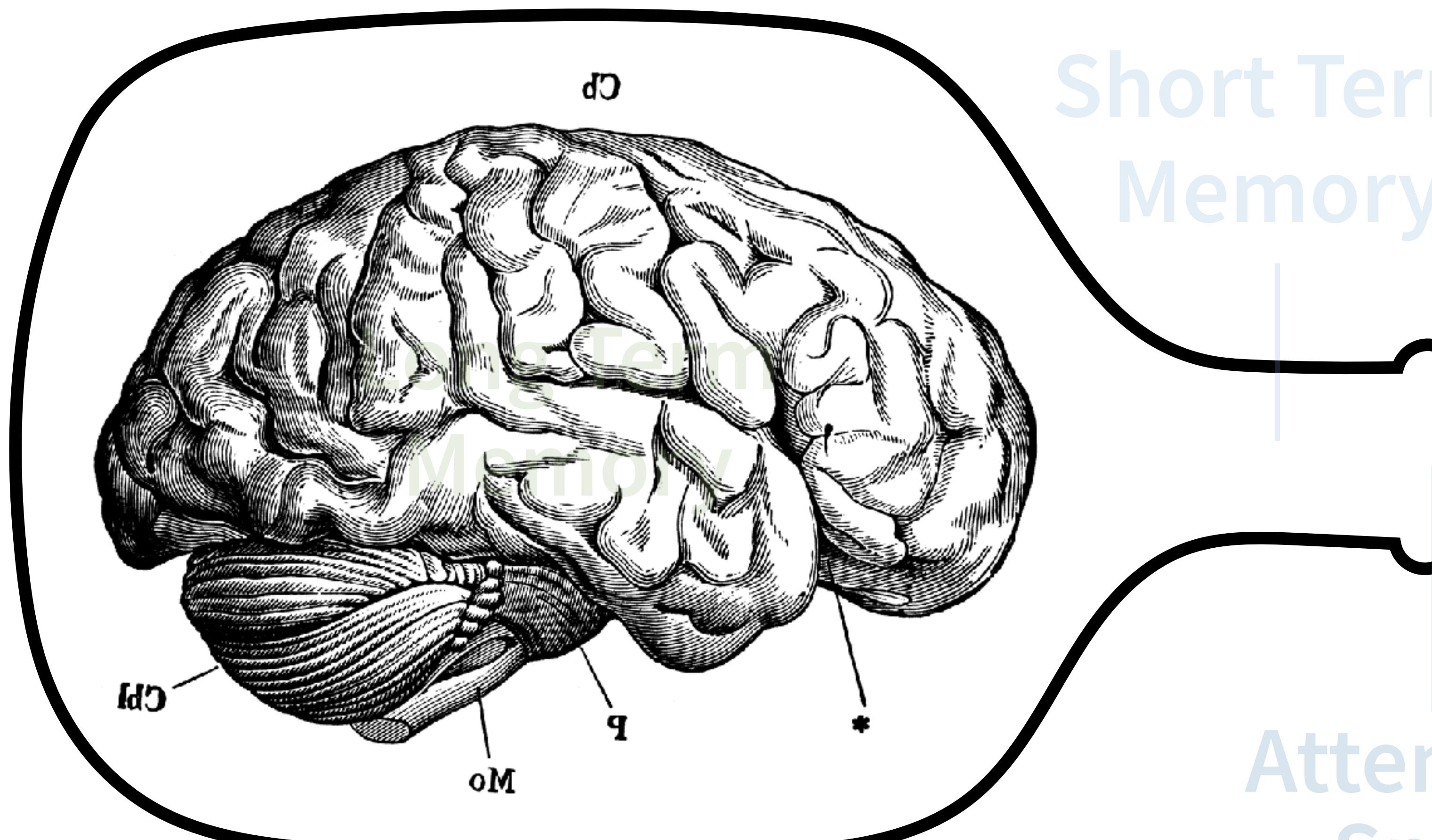
Henry L. Roediger III

Mark A. McDaniel



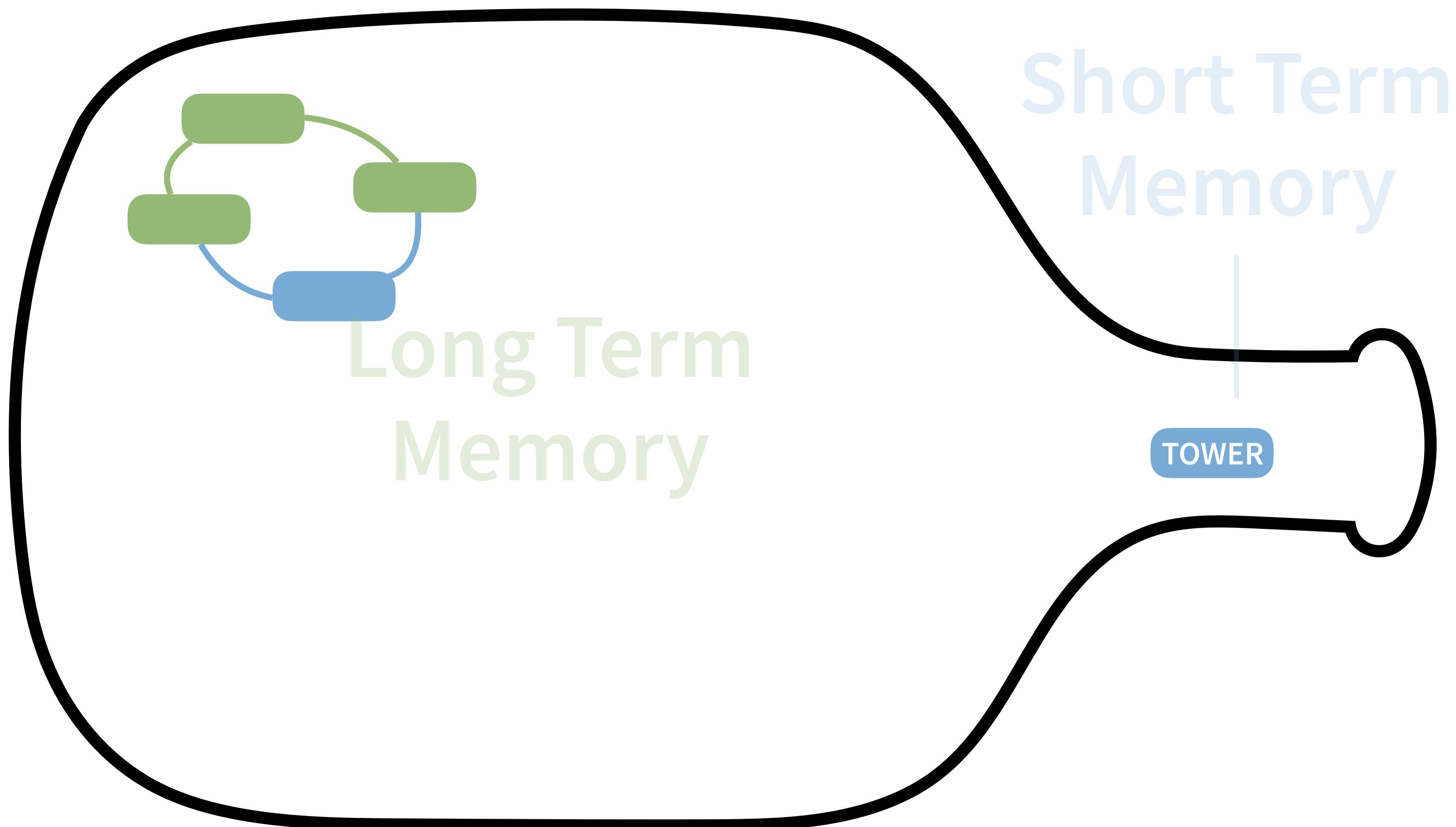


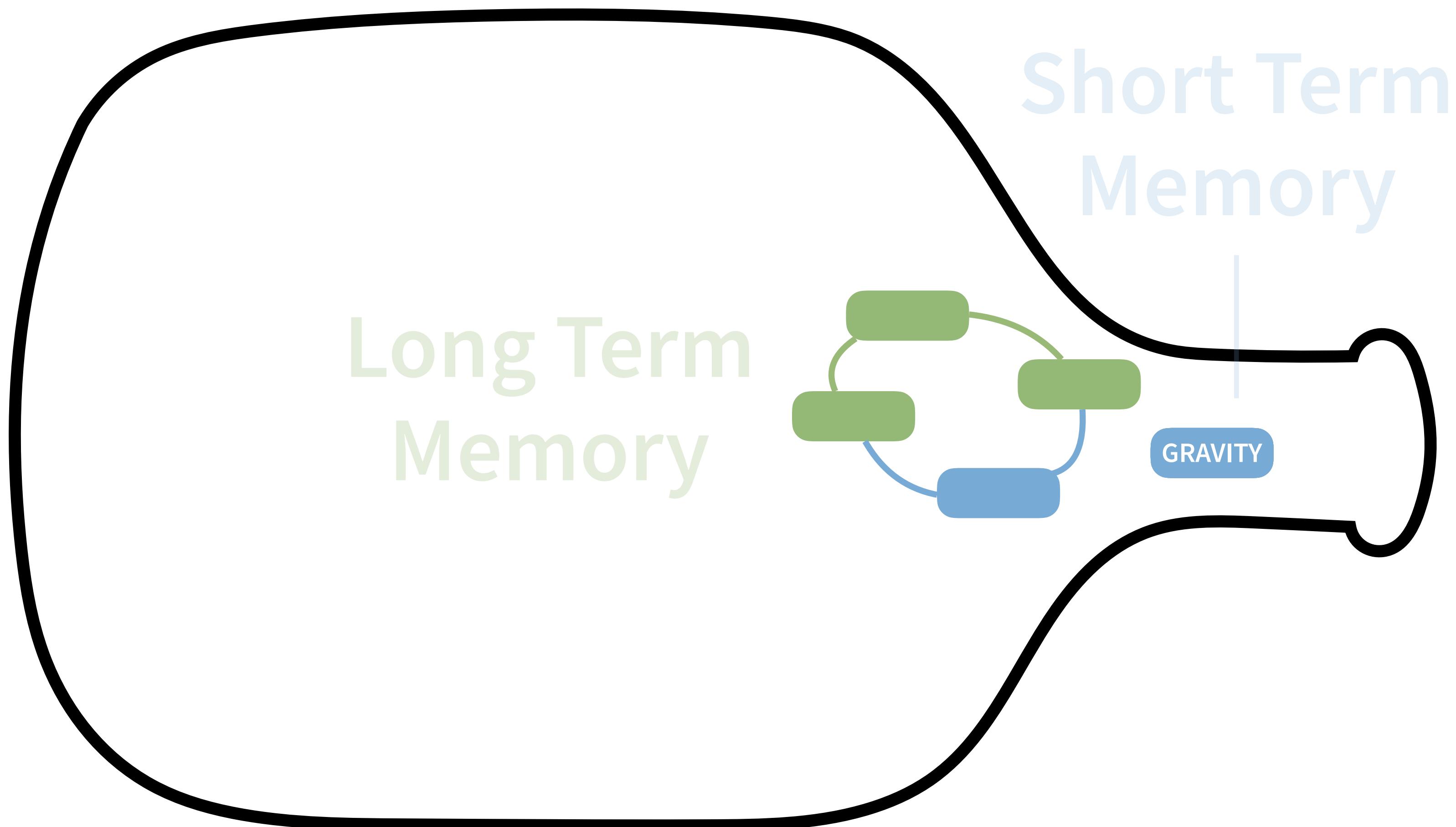


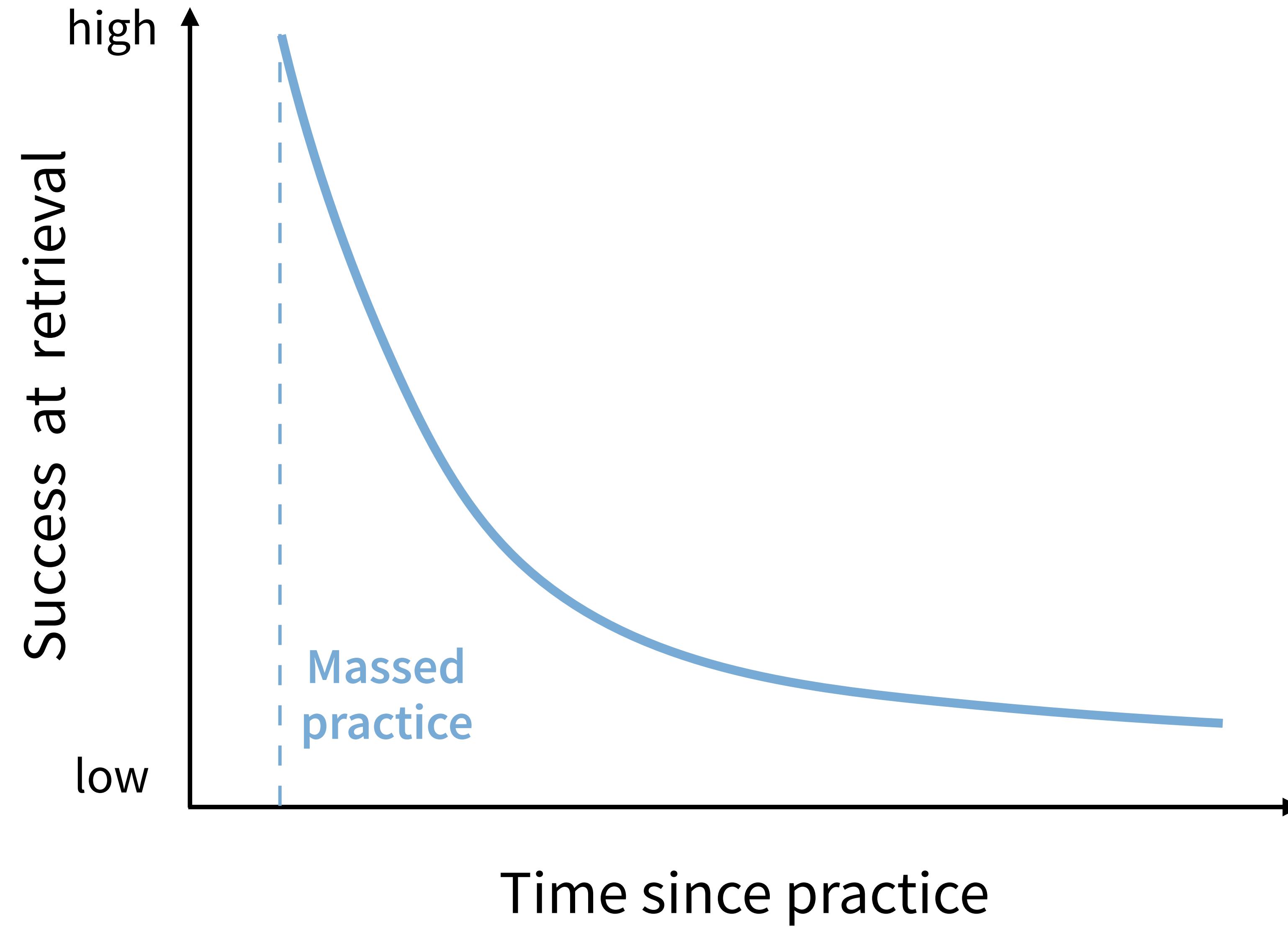


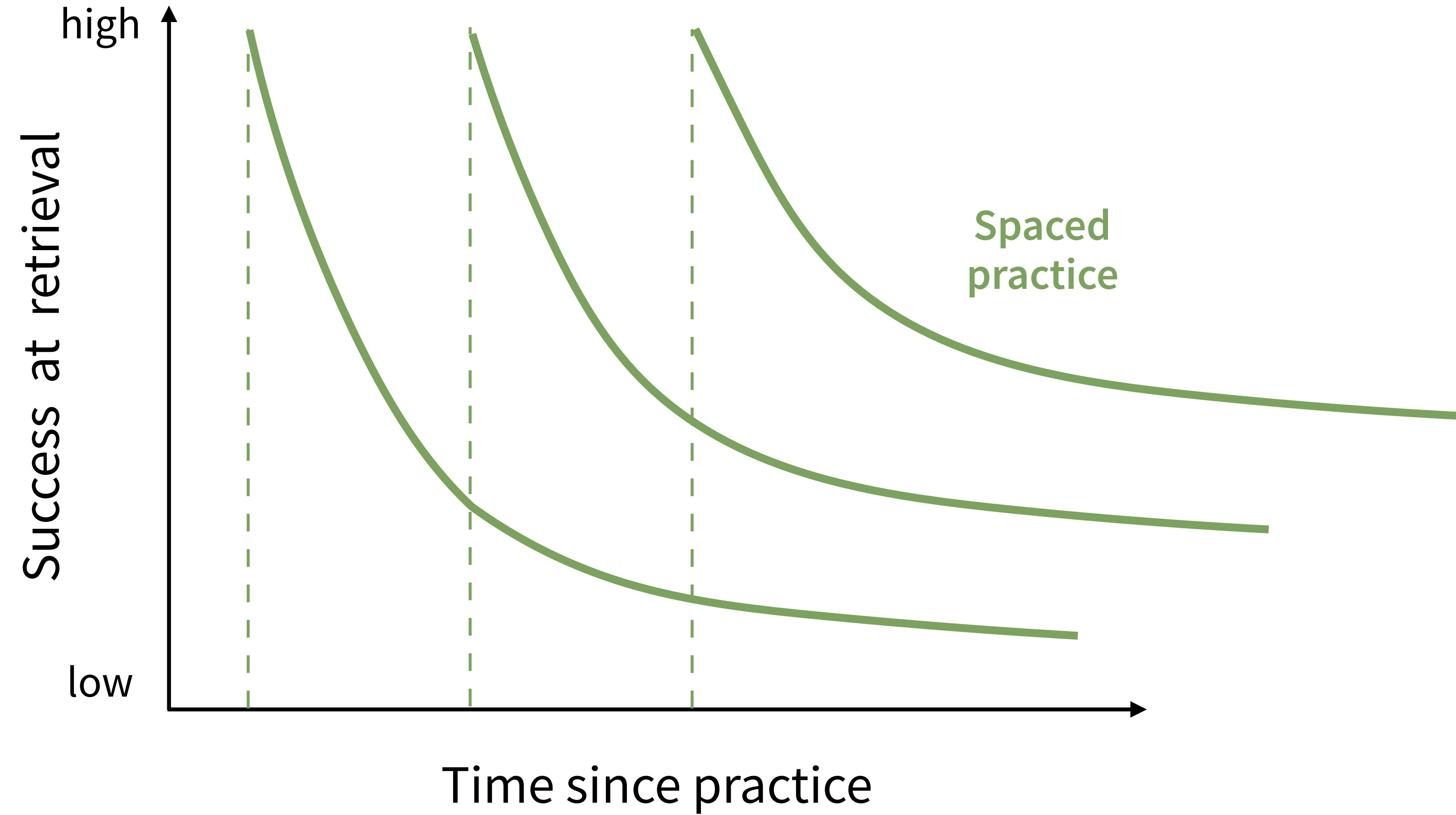
Short Term
Memory

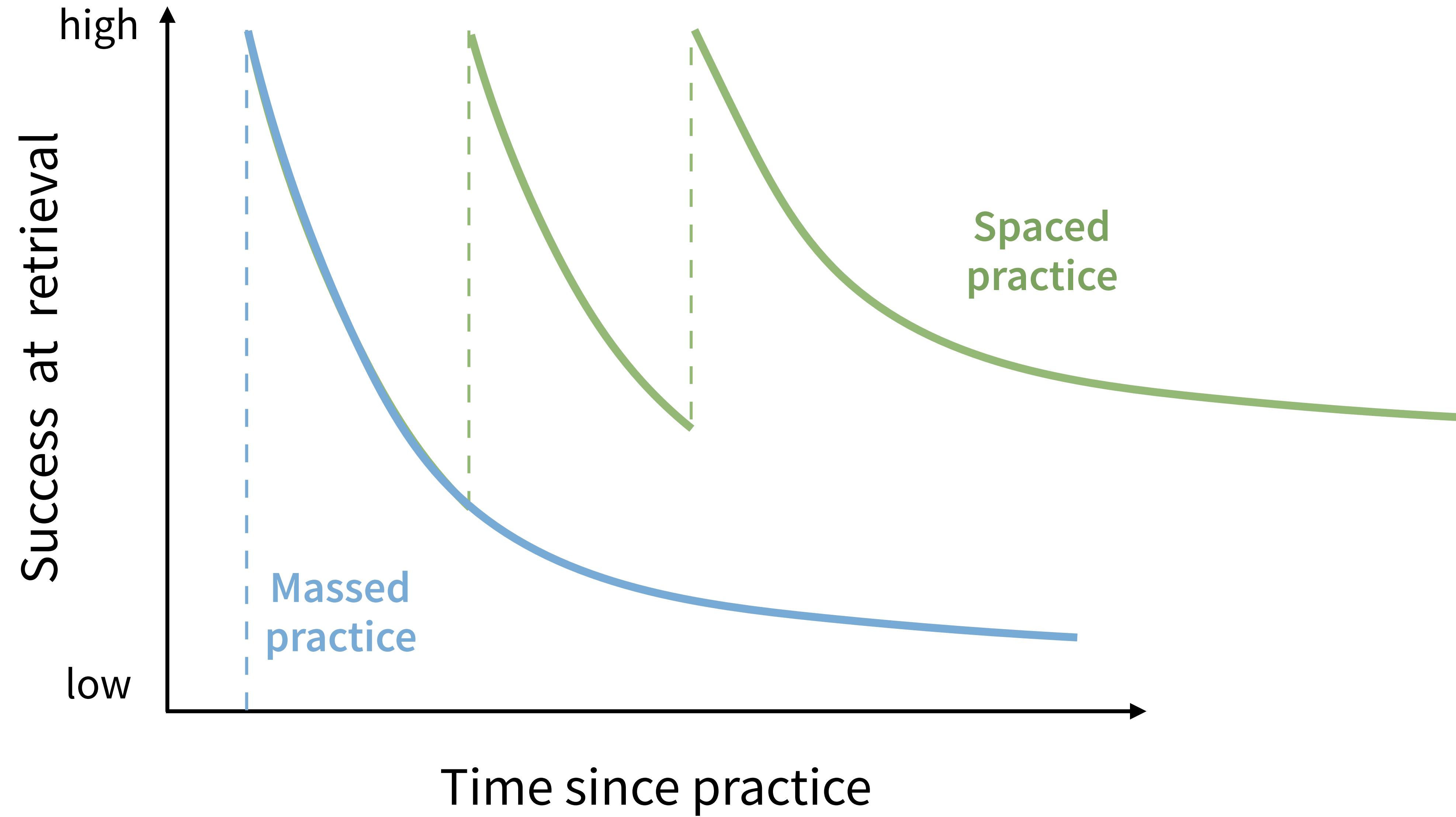
Attention
Span











How to teach a workshop

1. Teach the **mental model(s)**

Explain them. Practice the skills they support.

2. Put useful details in a **handout**

3. Follow up with **post-workshop quizzes**

Use cognitive theory as a guide

Your Turn

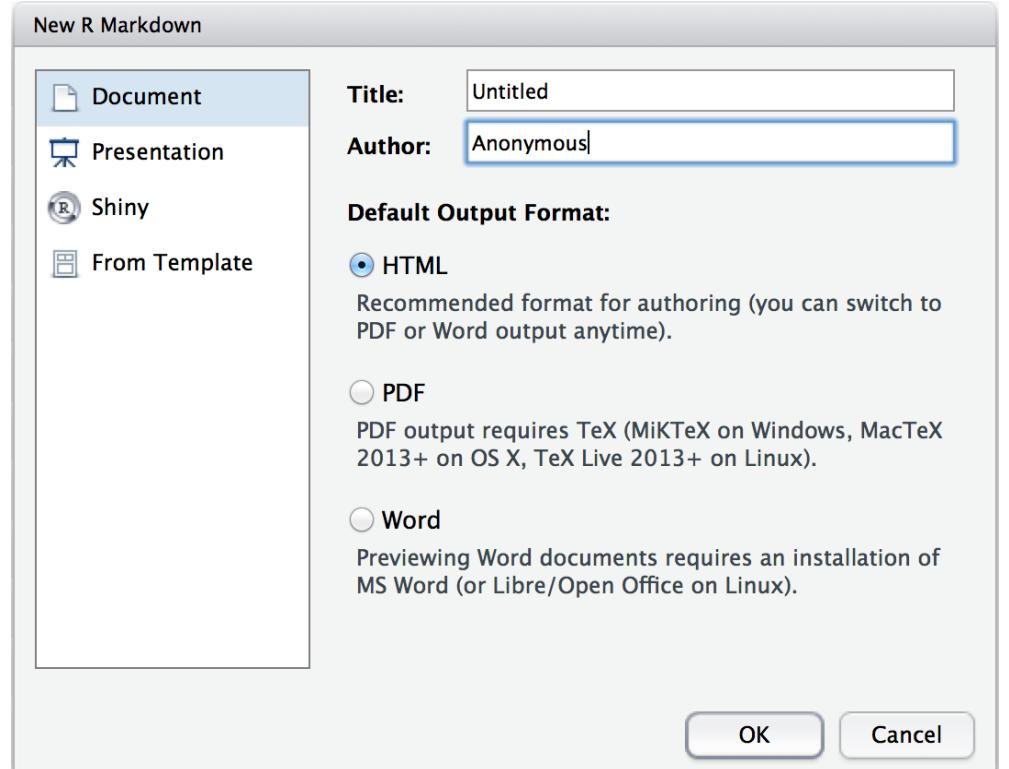
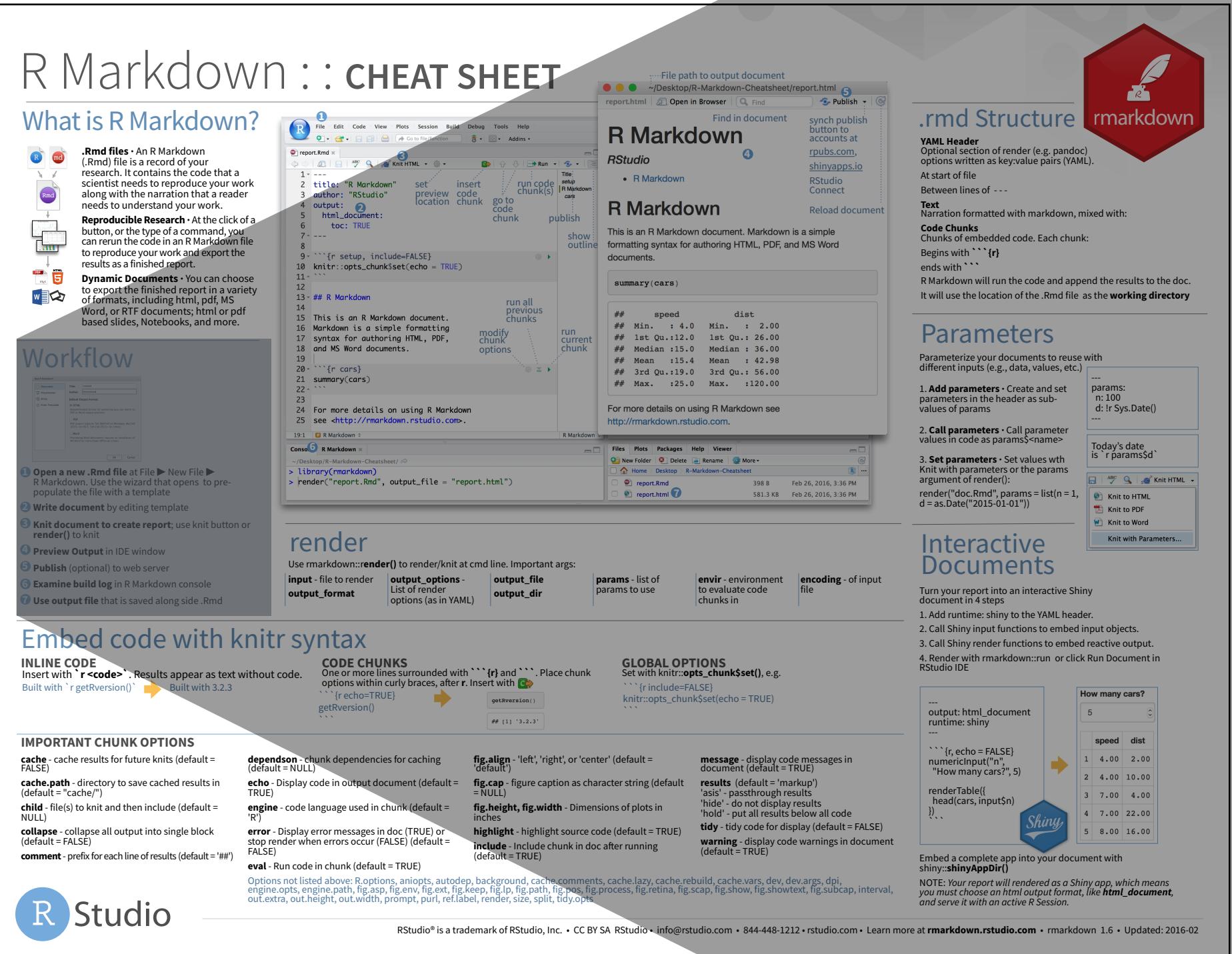
Type out three multiple choice questions about your topic and one coding exercise to test whether students have retained your learning objective(s).



R Markdown

**What is
R Markdown?**

Workflow



- 1 Open a new .Rmd file at File ► New File ► R Markdown. Use the wizard that opens to pre-populate the file with a template**
- 2 Write document by editing template**
- 3 Knit document to create report; use knit button or render() to knit**
- 4 Preview Output in IDE window**
- 5 Publish (optional) to web server**
- 6 Examine build log in R Markdown console**
- 7 Use output file that is saved along side .Rmd**

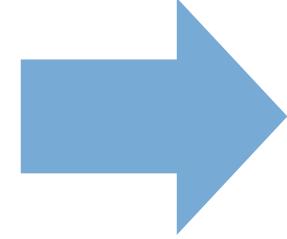
Your Turn

Create a New Project and open an R Markdown file.
Knit the file into finished HTML output. Read it.



Headers

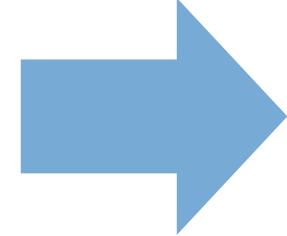
```
# Header 1  
## Header 2  
### Header 3  
#### Header 4  
##### Header 5  
##### Header 6
```



Header 1
Header 2
Header 3
Header 4
Header 5
Header 6

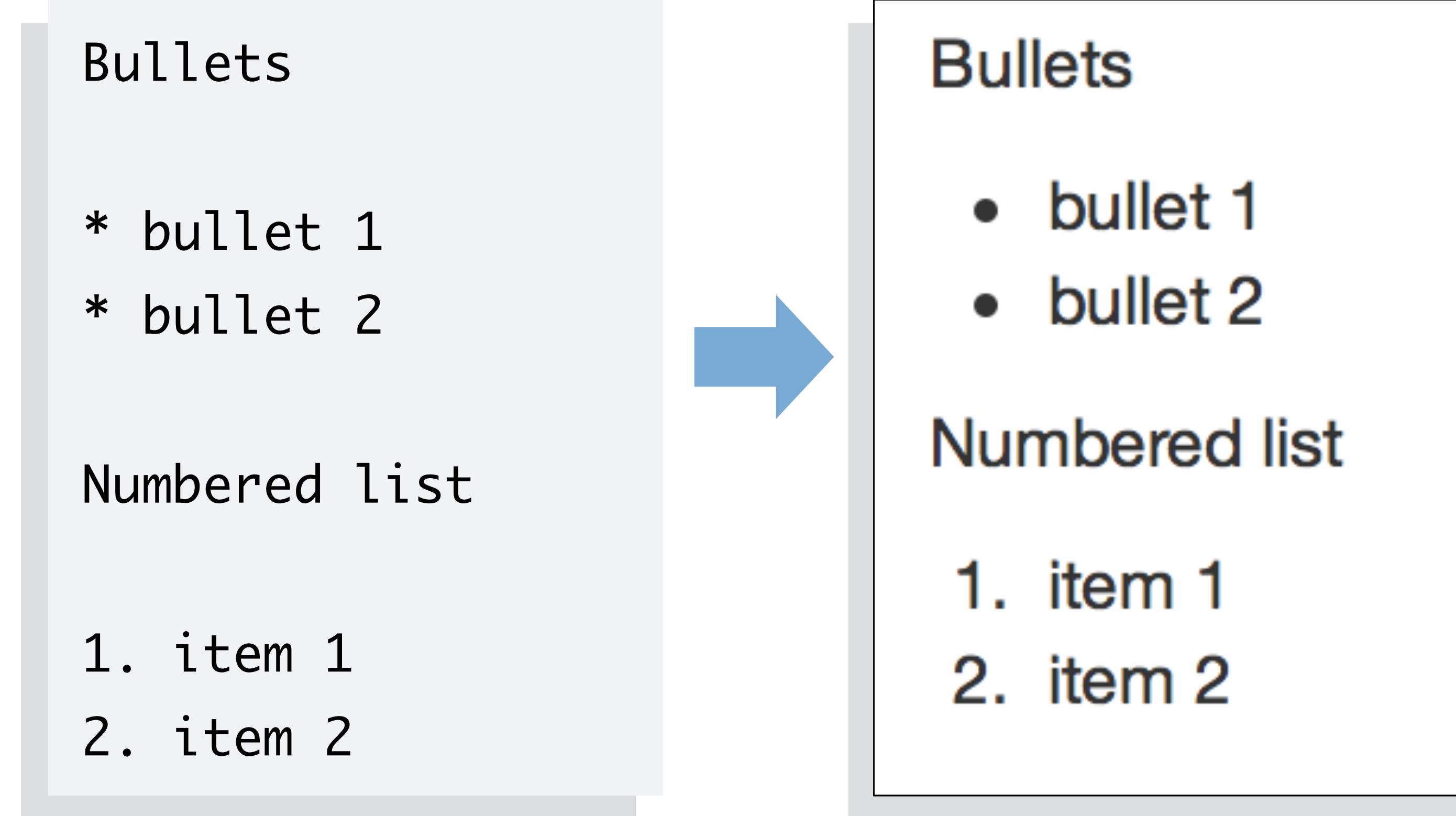
Text

```
Text  
_italics_  
__bold__  
'code'
```



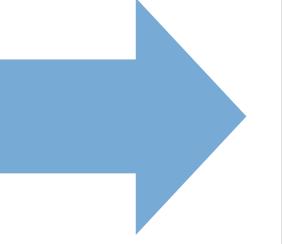
```
Text  
italics  
bold  
code
```

Lists



Hyperlinks

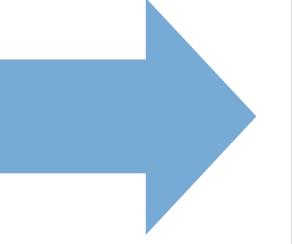
This is a
[link](www.git.com).



This is a link.

Equations

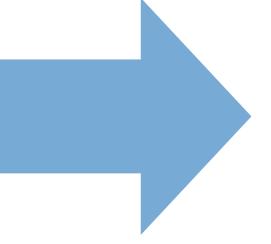
According to
Einstein,
 $E=mc^2$



According to
Einstein, $E = mc^2$

Images

The RStudio logo.



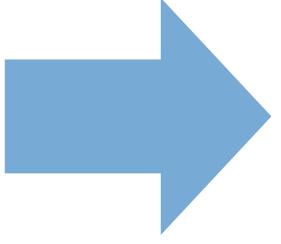
code chunks

A faint watermark of the R logo is visible in the bottom right corner, consisting of a circular arrow with the letters "R" inside.

Code chunks

Here's some code

```
```{r}  
dim(iris)
```
```

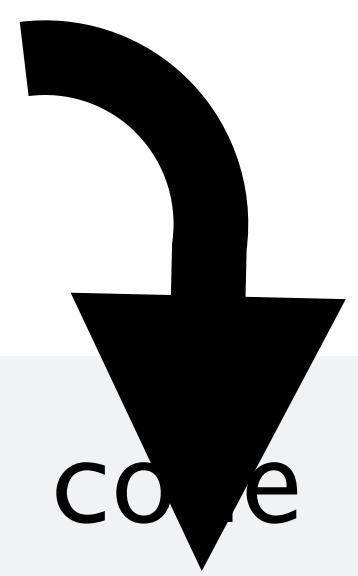


Here's some code

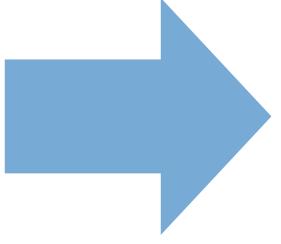
```
dim(iris)
```

```
## [1] 150 5
```

Chunk Options



```
Here's some code  
```{r echo=FALSE}  
dim(iris)
```
```

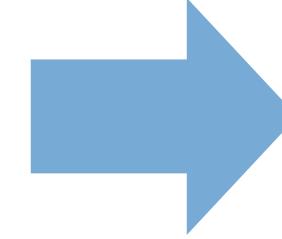


Here's some code

```
## [1] 150 5
```

Chunk Options - echo = FALSE

```
Here's some code  
```{r echo=FALSE}  
dim(iris)
```
```



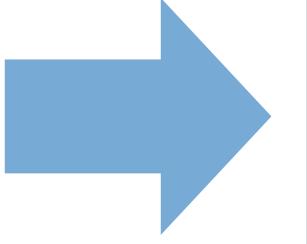
Here's some code

```
## [1] 150 5
```

Displays code results, but not code

Chunk Options - eval = FALSE

```
Here's some code  
```{r eval=FALSE}  
dim(iris)
```
```



```
Here's some code  
dim(iris)
```

Displays code, but not results (code is not run)

Your Turn

Use your R Markdown document to write a half page summary of your topic that includes bullet points and at least one chunk of example code.

Knit the document into an html file.



learnr



tutorials.shinyapps.io/02-separate-columns/

The screenshot shows a web-based Shiny application interface. The title bar reads "Separate and Unite Columns" and the URL is "https://tutorials.shinyapps.io/02-separate-columns/". The top right corner shows the name "Garrett".

The main content area has two columns. The left column, under the heading "Welcome", contains links: "separate()", "unite()", "Case study", and "Start Over". The "separate()" link is highlighted with a light gray background.

The right column starts with the heading "separate()". Below it is the word "hurricanes". A text block explains: "The `hurricanes` data set contains historical information about five hurricanes. At first glance it appears to contain four variables: `name`, `wind_speed`, `pressure`, and `date`. However, there are three more variables hidden in plain sight. Can you spot them?"

Below this text is a table showing the "hurricanes" data set:

| <code>name</code>

<code><chr></code> | <code>wind_speed</code>

<code><dbl></code> | <code>pressure</code> <code>date</code>

<code><dbl> <chr></code> |
|---|---|---|
| Alberto | 110 | 1007 2000-08-03 |
| Alex | 45 | 1009 1998-07-27 |
| Allison | 65 | 1005 1995-06-03 |
| Ana | 40 | 1013 1997-06-30 |
| Arlene | 50 | 1010 1999-06-11 |
| Arthur | 45 | 1010 1996-06-17 |

Text at the bottom of the table says "6 rows".

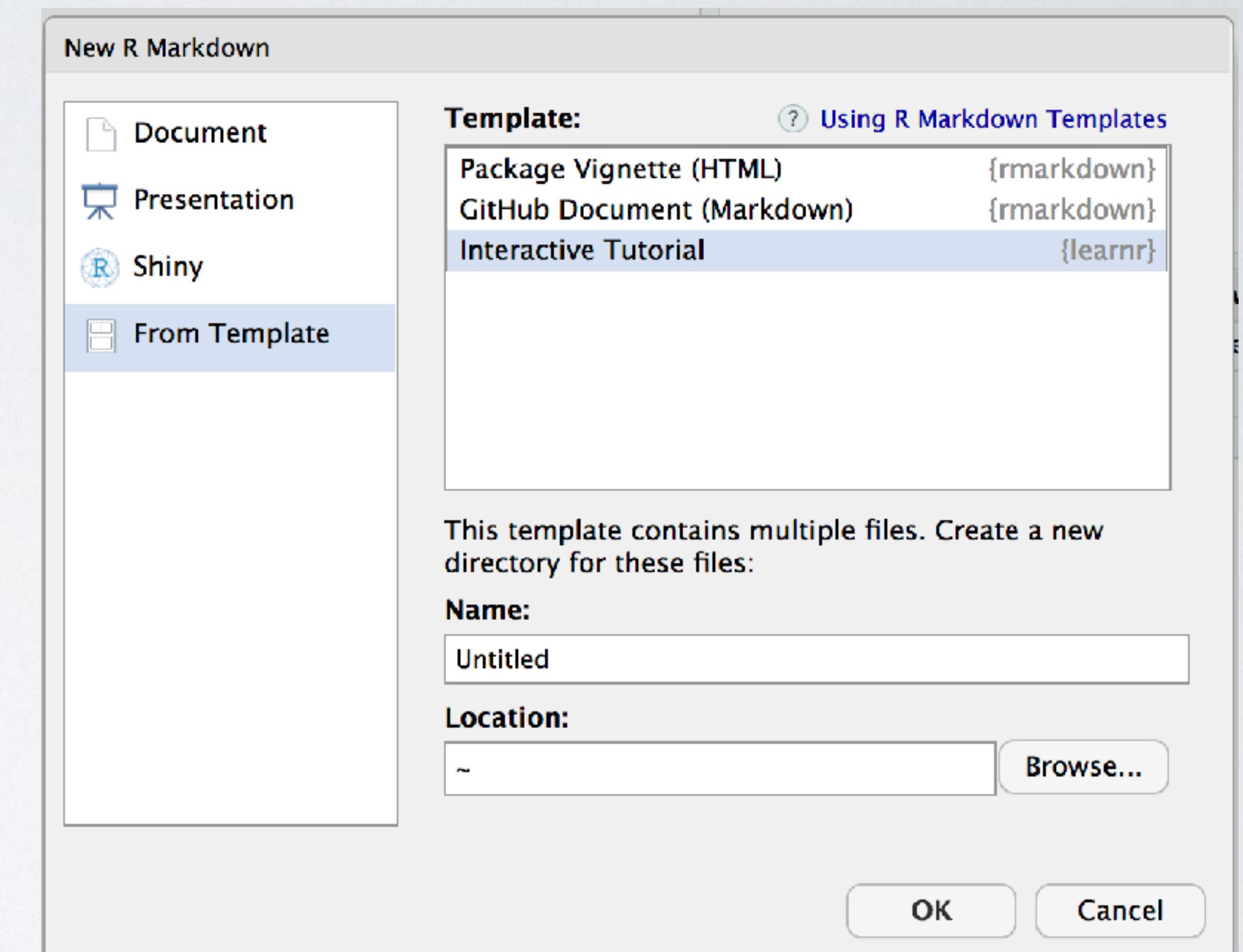
Below the table is a question: "Which variables are "hidden" in hurricanes? Check three." It lists four options: "location", "year", "month", and "day", each preceded by an empty checkbox.

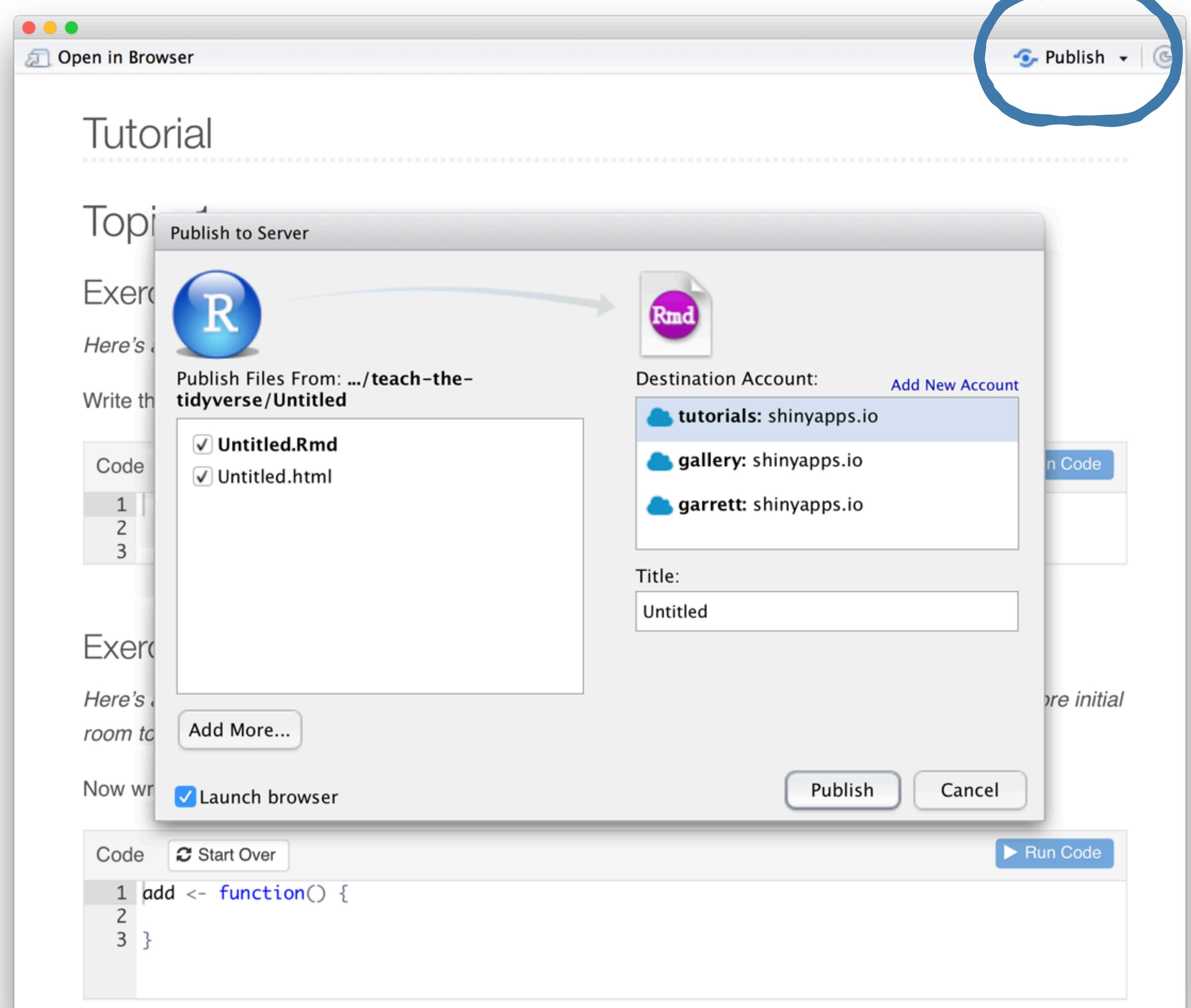
A blue "Submit Answer" button is located at the bottom of this section.

Your Turn

In your Teach the Tidyverse Project (rstudio.cloud/project/11787). Open a new R Markdown file and select From Template > Interactive Tutorial.

Read the source code in the file and run the file.





Learner setup

```
---
```

```
title: "Tutorial"
output: learnr::tutorial
runtime: shiny_prerendered
```

```
---
```

```
```{r setup, include=FALSE}
library(learnr)
```
```

Learner setup

```
---
```

```
title: "Tutorial"
```

```
output: learnr::tutorial
```

```
runtime: shiny_prerendered
```

```
---
```

```
```{r setup, include=FALSE}
```

```
library(learnr)
```

```
```
```

Set output to
learnr:: tutorial

Set runtime to
shiny_prerendered

Load the learnr
functions

Multiple choice questions

```
```{r q1}
question("Which package makes graphs?",
 answer("tidyr"),
 answer("dplyr"),
 answer("ggplot2", correct = TRUE),
 answer("purrr")
)
```
```

Multiple choice questions

question()

```
question("Which package makes graphs?",  
        answer("tidyr"),  
        answer("dplyr"),  
        answer("ggplot2", correct = TRUE),  
        answer("purrr"))
```

)

```

# Multiple choice questions

question()

prompt

```
question("Which package makes graphs?",
 answer("tidyr"),
 answer("dplyr"),
 answer("ggplot2", correct = TRUE),
 answer("purrr"))
```

)

...

# Multiple choice questions

question()

prompt

```
question("Which package makes graphs?",
 answer("tidyr"),
 answer("dplyr"),
 answer("ggplot2", correct = TRUE),
 answer("purrr"))
```

answers

# Multiple choice questions

question()

prompt

```
question("Which package makes graphs?",
 answer("tidyr"),
 answer("dplyr"),
 answer("ggplot2", correct = TRUE),
 answer("purrr"))
```

answers

correct answer

# Multiple choice questions

Use **incorrect** to return failure message

```
```{r q1}
question("Which package makes graphs?",
  answer("tidyr"),
  answer("dplyr"),
  answer("ggplot2", correct = TRUE),
  incorrect = "Try Again"
)
```

Multiple choice questions

Use **message** to return messages for specific items

```
```{r q1}
question("Which package makes graphs?",
 answer("tidyr"),
 answer("dplyr"),
 answer("ggplot2", correct = TRUE),
 answer("purrr", message = "purrr is for lists"))
```
```

Your Turn

1. Turn your R Markdown file into a learnr tutorial by placing in its yaml (then save):

output: learnr::tutorial

runtime: shiny_prerendered

2. In a new `` ` chunk add the first of your multiple choice questions.

3. Run the document and try the question (you may need to click through to open in browser)



Quizzes

Combine several questions with **quiz()**

```
```{r quiz}
quiz(caption = "Pop Quiz",
 question("Which package makes graphs?",
 answer("dplyr"),
 answer("ggplot2", correct = TRUE)),
 question("Doesforcats work with factors?",
 answer("yes", correct = TRUE), answer("no"))
)
```
```

Quizzes

Combine several questions with **quiz()**

```
```{r quiz}

quiz(caption = "Pop Quiz",
 question("Which package makes graphs?",
 answer("dplyr"),
 answer("ggplot2", correct = TRUE)),
 question("Doesforcats work with factors?",
 answer("yes", correct = TRUE), answer("no"))

)
```
```

Your Turn

Add your last two multiple choice questions to the first *as a quiz*.

Run your document and take the quiz.



Coding Exercises

```
```{r ex1, exercise=TRUE}  
mtcars
```
```

Coding Exercises

```
```{r ex1, exercise=TRUE}  
mtcars
```
```

Must have
chunk label

Coding Exercises

```
```{r ex1, exercise=TRUE}  
mtcars
```
```

Must have
chunk label

Must have
exercise = TRUE

Coding Exercises

```
```{r ex1, exercise=TRUE}
```

`mtcars`

...

Optional code  
to pre-populate

Must have  
chunk label

Must have  
`exercise = TRUE`

# Coding Exercises

Add **exercise.eval=TRUE** to run initial code

```
```{r ex1, exercise=TRUE, exercise.eval=TRUE}
mtcars
```
```

# Coding Exercises

Code in **<label>-setup** will be run before students code

```
```{r ex1, exercise=TRUE}  
mtcars  
```
```

```
```{r ex1-setup}  
mtcars <- mtcars[1:10, ]  
```
```

# Coding Exercises

Code in **<label>-setup** will be run before students code

```
\`{r ex1, exercise=TRUE}
mtcars
\`
```

```
\`{r ex1-setup}
mtcars <- mtcars[1:10,]
\`
```

# Coding Exercises

Code in **<label>-solution** will be available as a solution

```
```{r ex1, exercise=TRUE}  
mtcars  
```
```

```
```{r ex1-solution}  
head(mtcars)  
```
```

# Coding Exercises

Code in **<label>-solution** will be available as a solution

```
\`{r ex1, exercise=TRUE}
mtcars
\`
```

```
\`{r ex1-solution}
head(mtcars)
\`
```

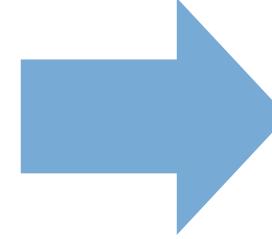
# Your Turn

Add your coding exercise to your learnr document as well as a solution. Test it out.



# Headers

```
Header 1
Header 2
Header 3
Header 4
Header 5
Header 6
```



**Header 1**  
**Header 2**  
**Header 3**  
**Header 4**  
**Header 5**  
**Header 6**



← → ⌂ ⌄

Secure | https://garrett.rstudio.cloud/8c34a197a84f46528d76aa11c8177deb/p/7519/Untitled.Rmd#section-topic-1

# Tutorial

Topic 1

Topic 2

Start Over

## Title

Topic 1

Exercise

*Here's a simple exercise with an empty code chunk provided for entering the answer.*

Write the R code required to add two plus two:

Code    Start Over

```
1
2
3
```

## Exercise with Code

*Here's an exercise with some prepopulated code as well as `exercise.lines = 5` to provide a bit more space.*

Now write a function that adds any two numbers and then call it:

RStudio Cloud Tutorial

Secure | https://garrett.rstudio.cloud/8c34a197a84f46528d76aa11c8177deb/p/7915/Untitled.Rmd#section-topic-1

# Tutorial

Second Level Header

Second Level Header

Start Over

## First Level Header (largely ignored)

Second Level Header

Third Level Header

*Here's a simple exercise with an empty code chunk provided for entering the answer.*

Write the R code required to add two plus two:

Code    Start Over

```
1
2
3
```

Third Level Header

*Here's an exercise with some prepopulated code as well as `exercise.lines = 5` to provide a bit more space.*

Now write a function that adds any two numbers and then call it:

RStudio Cloud Tutorial

Secure | https://garrett.rstudio.cloud/8c34a197a84f46528d76aa11c8177deb/p/3102/Untitled.Rmd#section-topic-1

# Tutorial

Second Level Header

Second Level Header

Start Over

## First Level Header (largely ignored)

Second Level Header

Third Level Header

*Here's a simple exercise with an empty code chunk provided for entering the answer.*

Write the R code required to add two plus two:

Code Start Over

1  
2  
3

Continue

# Progressive Reveal

Students will need to click "Continue" to move on to each third level header (###)

```

```

```
title: "Tutorial"
```

```
output:
```

```
 learnr::tutorial:
```

```
 progressive: true
```

```
runtime: shiny_prerendered
```

```

```

new line + indent two  
spaces + colon to add  
options to learnr::tutorial

indent four spaces

# Progressive Reveal

Students will need to click "Continue" to move on to each third level header (###)

```

```

```
title: "Tutorial"
```

```
output:
```

```
 learnr::tutorial:
```

```
 progressive: true
```

```
runtime: shiny_prerendered
```

```

```

new line + indent two  
spaces + colon to add  
options to learnr::tutorial

indent four spaces

# Allow Skip

Add **allow\_skip: true** to let students continue without finishing each exercise.

```

```

```
title: "Tutorial"
output:
 learnr::tutorial:
 progressive: true
 allow_skip: true
runtime: shiny_prerendered
```

# [rstudio.github.io/learnr/](https://rstudio.github.io/learnr/)

The screenshot shows a web browser window titled "Interactive Tutorials for R" with the URL "https://rstudio.github.io/learnr/". The page has a blue header with tabs for "learnr", "Home", "Exercises", "Questions", "Publishing", "Formats", and "Examples". A sidebar on the left is titled "Overview" and lists "Getting Started", "Tutorial Types", "Exercises", "Questions", "Videos", "Shiny Components", "External Resources", "Preserving Work", and "Publishing". The main content area features a large title "Interactive Tutorials for R" and a sub-section "Overview". It explains that the **learnr** package makes it easy to turn any [R Markdown](#) document into an interactive tutorial. Tutorials consist of content along with interactive components for checking and reinforcing understanding. Tutorials can include any or all of the following:

1. Narrative, figures, illustrations, and equations.
2. Code exercises (R code chunks that users can edit and execute directly).
3. Quiz questions.
4. Videos (supported services include YouTube and Vimeo).
5. Interactive Shiny components.

Tutorials automatically preserve work done within them, so if a user works on a few exercises or questions and returns to the tutorial later they can pick up right where they left off.

## Examples

Here are some simple examples of tutorials created with the **learnr** package:

This screenshot shows a "Data basics" tutorial. It includes a section on "Tables" with a question "What is a table?". Below the question is a code chunk that generates a table of US state abbreviations and their populations. The table is displayed in a light gray grid format.

This screenshot shows a "Filter observations" tutorial. It includes a section on "Filter rows with filter()". Below the section is a code chunk that filters a dataset of flight information based on departure time. The resulting table shows filtered rows for flights departing at 10:00 AM and 11:00 AM.

This screenshot shows a "Summarise Tacles" tutorial. It includes a section on "Combining multiple operations". Below the section is a code chunk that performs multiple operations on a dataset. The result is a scatter plot showing the relationship between "ArrTime" and "DepTime" for various flights.

# Your Turn

Add `progressive: true` and `allow_skip:` to the yaml of the document. Then insert second and third level headers as needed to make a satisfying navigation experience.

Work through your complete tutorial.



# Thank You



# Thank You



Please take the class survey  
[www.surveymonkey.com/r/9L6PTBF](https://www.surveymonkey.com/r/9L6PTBF)