# semana 4

Luis Ambrocio

24/8/2021

## Contenido

**Swirl** **1**
escribiendo cursos swirl: . . . 2
Lecciones de escritura: tipos de preguntas . . . 3
Lecciones de escritura: tipos de preguntas . . . 3
Lecciones de escritura: la meta pregunta . . . 3
Lecciones de escritura: Preguntas sobre mensajes . . . 3
Lecciones de escritura: preguntas de comando . . . 4
Preguntas de respuestas múltiples . . . 4
Otros tipos de preguntas . . . 5
Organizar lecciones . . . 5
Compartiendo tu curso . . . 5
Compartir su curso como un archivo . . . 5
Compartiendo su curso en GitHub . . . 6
Compartiendo su curso en The Swirl Course Network . . . 6
Más ayuda y recursos . . . 6

## Swirl

¿Qué es el swirl? Parte 1

- Swirl es un paquete R que convierte la consola R en un entorno de aprendizaje interactivo.
- Los estudiantes son guiados a través de ejercicios de programación de R donde pueden responder preguntas en la consola de R.
- No hay separación entre dónde un estudiante aprende a usar R y dónde pasa a usar R para sus propios propósitos creativos.

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

- ¡Cualquiera puede crear y distribuir su propio curso de swirl!
- Crear un curso en swirl puede permitirle escalar la educación en ciencia de datos y R dentro de su empresa o escuela.
- Si está interesado en participar en la comunidad educativa de código abierto, puede lanzar su curso swirl en línea y Team swirl lo promoverá por usted.
- Hay un paquete llamado swirlify que está diseñado para ayudarte a escribir un curso de swirlify. Veremos en swirlify más tarde.

¿Qué es un curso swirl? - Un curso de swirl es un directorio que contiene todos los archivos, carpetas y lecciones asociadas con el curso que está desarrollando. - Las lecciones contienen la mayor parte del contenido con el que interactúan los estudiantes y el curso organiza estas lecciones de manera secuencial. -

Un curso debe encapsular conceptualmente un concepto amplio que desea enseñar. Por ejemplo: “Trazar en R” o “Estadísticas para ingenieros” son temas lo suficientemente amplios como para desglosarlos más (en lecciones de las que hablaremos a continuación). - Cuando un estudiante comienza a girar, se le pedirá que elija de una lista de cursos, y luego puede elegir una lección dentro del curso que seleccionó. - Una vez que el estudiante selecciona una lección, swirl comenzará a preguntarle preguntas en la consola R.

¿Qué es una lección?

- Una lección es un directorio que contiene todos los archivos necesarios para ejecutar una unidad de instrucción dentro de swirl.
- Por ejemplo, un curso de “Trazado en R” puede contener las lecciones: “Trazado con gráficos base”, “Trazado con celosía” y “Trazado con ggplot2”.
- Cada lección debe contener un archivo `Lesson.yaml` que estructura el texto que el alumno verá dentro de la consola R mientras usa Swirl.
- El archivo `Lesson.yaml` contiene una secuencia de preguntas que se les pedirá a los estudiantes.

primero:

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

Luego, configure su directorio de trabajo donde desee crear su curso e inicie la aplicación de creación de lecciones Shiny:

```
setwd(file.path("~", "Desktop"))
swirlify("Lesson 1", "My First Course")
```

## escribiendo cursos swirl:

uso de la aplicación

Para ver una demostración sobre cómo usar la aplicación de creación Shiny, vea el siguiente video: https://youtu.be/UPL_W-Umgjs

usando la consola R

- Alternativamente, puede usar la consola R y un editor de texto para escribir lecciones.
- Recomendamos encarecidamente usar RStudio para escribir lecciones de swirl, ya que RStudio proporciona este editor y la configuración de la consola por defecto.

Para comenzar una nueva lección desde la consola R, establezca su directorio de trabajo donde desea crear el curso, y luego use la función `new_lesson ()` para crea el curso:

```
setwd("/Users/sean/")
new_lesson("My First Lesson", "My First Course")
```

### Part 1

La función `new_lesson ()` creará una estructura de archivo y carpeta como esta en su directorio de trabajo:

```
My_New_Course
- My_First_Lesson
    - lesson.yaml
    - initLesson.R
    - dependson.txt
    - customTests.R
```

Para revisar: la función `new_lesson ()` hizo lo siguiente:

1. Se creó un nuevo directorio en `/Userss/wirlsean/` llamado `My_New_Course`.
2. Se creó un nuevo directorio en `/ Userss/wirlseans/wirlMy_New_Course` llamado`My_First_Lesson`.

- Se crearon varios archivos dentro de `/Userss/wirlseans/wirlMy_New_Courses/wirlMy_First_Lesson`:
  - `Lesson.yaml` es donde escribirás todas las preguntas de esta lección. (Ejemplo)
  - `initLesson.R` es un script de R que se ejecuta antes de que comience la lección, que es generalmente se usa para cargar datos o configurar variables ambientales. (Ejemplo)
  - `dependson.txt` es la lista de paquetes R que requerirá su lección. swirl instalará estos paquetes si el usuario aún no los tiene instalados. (Ejemplo)
  - `customTests.R` es donde puede escribir sus propias pruebas para las respuestas de los estudiantes. (Ejemplo)

Si todo está configurado correctamente, entonces `new_lesson ()` debería haber abierto el nuevo archivo `Lesson.yaml` en un editor de texto. Ahora puede comenzar a agregar preguntas a la `lección.yaml` mediante el uso de funciones que comienzan con`wq_` (escriba la pregunta).

### Lecciones de escritura: tipos de preguntas

Las lecciones son secuencias de preguntas que tienen la siguiente estructura general:

```
- Class: [type of question]
  Key1: [value1]
  Key2: [value2]

- Class: [type of question]
  Key1: [value1]
  Key2: [value2]
...
```

El ejemplo anterior muestra la estructura de alto nivel para dos preguntas.

### Lecciones de escritura: tipos de preguntas

- Cada pregunta está delimitada con un guión.
- Cada pregunta comienza con una "Clase" que especifica el comportamiento de esa pregunta dentro del swirl.
- Lo que sigue a la clase es un conjunto de pares clave-valor que se usarán para representar la pregunta cuando un estudiante esté usando swirl.

### Lecciones de escritura: la meta pregunta

La primera pregunta en cada `lección.yaml` es siempre la meta pregunta que contiene información general sobre el curso. A continuación se muestra un ejemplo de la meta pregunta:

```
- Class: meta
  Course: My Course
  Lesson: My Lesson
  Author: Dr. Jennifer Bryan
  Type: Standard
  Organization: The University of British Columbia
  Version: 2.5
```

The meta question will not be displayed to a student. The only fields you should modify are `Author` and `Organization` fields.

### Lecciones de escritura: Preguntas sobre mensajes

Las preguntas de mensaje muestran una cadena de texto en la consola R para que el alumno la lea. Una vez que el estudiante presiona Enter, swirl pasará a la siguiente pregunta.

Agregue una pregunta de mensaje usando `wq_message()`.

Aquí hay una pregunta de mensaje de ejemplo:

```
- Class: text
  Output: Welcome to my first swirl course!
```

El alumno verá lo siguiente en la consola de R:

```
| Welcome to my first swirl course!

...
```

**Lecciones de escritura: preguntas de comando**

Las preguntas de comando le piden al alumno que escriba una expresión en la consola R.

- El `CorrectAnswer` se ingresa en la consola si el estudiante usa el`skip ()` función.
- La "Pista" se muestra al alumno si no responde correctamente la pregunta.
- Las `AnswerTests` determinan si el estudiante respondió o no a la pregunta correctamente. Consulte la sección de prueba de respuestas para obtener más información.

Agregue una pregunta de mensaje usando `wq_command ()`.

Aquí hay una pregunta de comando de ejemplo:

```
- Class: cmd_question
  Output: Add 2 and 2 together using the addition operator.
  CorrectAnswer: 2 + 2
  AnswerTests: omnitest(correctExpr='2 + 2')
  Hint: Just type 2 + 2.
```

el estudiante vera lo siguiente en la consola

```
| Add 2 and 2 together using the addition operator.

>
```

**Preguntas de respuestas múltiples**

Las preguntas de opción múltiple presentan una selección de opciones para el estudiante. Estas opciones se presentan en un orden diferente cada vez que se ve la pregunta.

- "AnswerChoices" debe ser una cadena de opciones separadas por punto y coma entre las que el estudiante tendrá que elegir.

Agregue una pregunta de mensaje usando `wq_multiple()`.

Aquí hay un ejemplo de pregunta de opción múltiple:

```
- Class: mult_question
  Output: What is the capital of Canada?
  AnswerChoices: Toronto;Montreal;Ottawa;Vancouver
  CorrectAnswer: Ottawa
  AnswerTests: omnitest(correctVal='Ottawa')
  Hint: This city contains the Rideau Canal.
```

El alumno vera lo siguiente

```
| What is the capital of Canada?

1: Toronto
2: Montreal
3: Ottawa
```

```
4: Vancouver
```

### Otros tipos de preguntas

Para obtener documentación completa sobre cómo escribir cursos y lecciones de swirlify, visite el sitio web de swirlify: http://swirlstats.com/swirlify/

## Organizar lecciones

Repasemos la estructura general de un recorrido en espiral. Esta es la estructura de un curso con dos lecciones:

```
My_New_Course
- My_First_Lesson
  - lesson.yaml
  - initLesson.R
  - dependson.txt
  - customTests.R
- My_Second_Lesson
  - lesson.yaml
  - initLesson.R
  - dependson.txt
  - customTests.R
```

- De forma predeterminada, cada carpeta en `My_New_Course` se mostrará al estudiante como una lección que puede seleccionar.
- Si desea especificar explícitamente el orden en el que se muestran las lecciones, deberá agregar un archivo `MANIFEST` a su curso.
- Puede hacer esto con la función `add_to_manifest ()`, que agregará la lección en la que está trabajando actualmente al `MANIFEST`. También puede editar el "MANIFIESTO" usted mismo en un editor de texto.

El archivo (abreviado) `MANIFEST` a continuación pertenece al curso de Programación R de Team swirl:

```
Basic_Building_Blocks
Workspace_and_Files
Sequences_of_Numbers
Vectors
Missing_Values
Subsetting_Vectors
Matrices_and_Data_Frames
```

## Compartiendo tu curso

Swirlify hace que compartir un curso de swirl sea fácil. Recomendamos tres métodos diferentes para compartir un curso de swirl.

### Compartir su curso como un archivo

Hemos desarrollado el tipo de archivo `.swc` para que pueda compartir su curso como un solo archivo. Crear un archivo `.swc` para su curso es fácil:

1. Establezca cualquier lección del curso que desee compartir como la lección actual usando `set_lesson ()`.
2. Cree un archivo `.swc` usando la función`pack_course ()`. Su archivo `.swc` aparecerá en el mismo directorio que el directorio que contiene la carpeta del curso. También tiene la opción de exportar el archivo `.swc` a otro directorio especificando el argumento`export_path`.

- Ahora puede compartir su archivo `.swc` como lo haría con cualquier otro archivo (a través de correo electrónico, servicios de intercambio de archivos, etc.).
- Los estudiantes pueden instalar su curso desde el archivo `.swc` descargando el archivo y luego usando la función`install_course ()`en espiral, que les pedirá que seleccionen interactivamente el archivo que descargaron.

**Compartiendo su curso en GitHub**

- Le recomendamos encarecidamente que desarrolle su curso en GitHub para que podamos brindarle un mejor apoyo si tiene preguntas o necesita ayuda mientras desarrolla su curso.
- Desarrollar su curso en GitHub proporciona el beneficio adicional de que su curso estará instantáneamente listo para distribuirse.
- Los estudiantes pueden instalar su curso desde swirl usando la función `install_course_github ()`.
- Asegúrese de que el directorio de su curso sea la carpeta raíz de su repositorio de git. Para ver ejemplos de cursos que se han compartido en GitHub, puede explorar algunos de los cursos en Swirl Course Network.

**Compartiendo su curso en The Swirl Course Network**

El objetivo de Swirl Course Network es enumerar y organizar todos cursos de remolino disponibles. Visite la [página de inicio] (http://swirlstats.com/scn/) delSCN para obtener más información.

Después de agregar su curso a SCN, los estudiantes podrán instalar su curso usando `install_course ("[Nombre de su curso]")` en forma de remolino.

Para agregar su curso al SCN:

1. Cree un archivo `.swc` para su curso.
2. Bifurque https://github.com/swirldev/scn en GitHub.
3. Agregue el archivo `.swc` a su bifurcación.
4. Agregue un archivo Rmd a su bifurcación como este. Puede incluir una descripción de su curso, autores, un sitio web del curso y cómo instalar su curso.
5. Ejecute `rmarkdown :: render_site ()` cuando su directorio actual esté configurado en su tenedor.
6. Agregue, confirme y envíe sus cambios a GitHub, luego envíenos una solicitud de extracción.

## Más ayuda y recursos

- El sitio web de Swirl
- La documentación de swirlify
- The Swirl Course Network

No dude en ponerse en contacto con Team Swirl:

- Por correo electrónico: info@swirlstats.com
- En Twitter: @ [swirlstats] (https://twitter.com/swirlstats)