



Appsilon

Optimización del uso de datos Shiny



Introducción práctica

Oriol Senan Programador R/Shiny

oriol@appsilon.com

Contenidos

- Introducción
- Herramientas para el análisis de performance
- Cargar y manejar datos en Shiny
- Conclusiones
- Actividad Práctica

Performance y Shiny



Tiempo de cálculo

89%

Uso de memoria

Performance y Shiny



Tiempo de cálculo



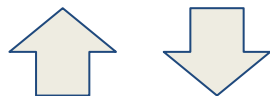
89%

Uso de memoria

Performance y Shiny



Tiempo de cálculo



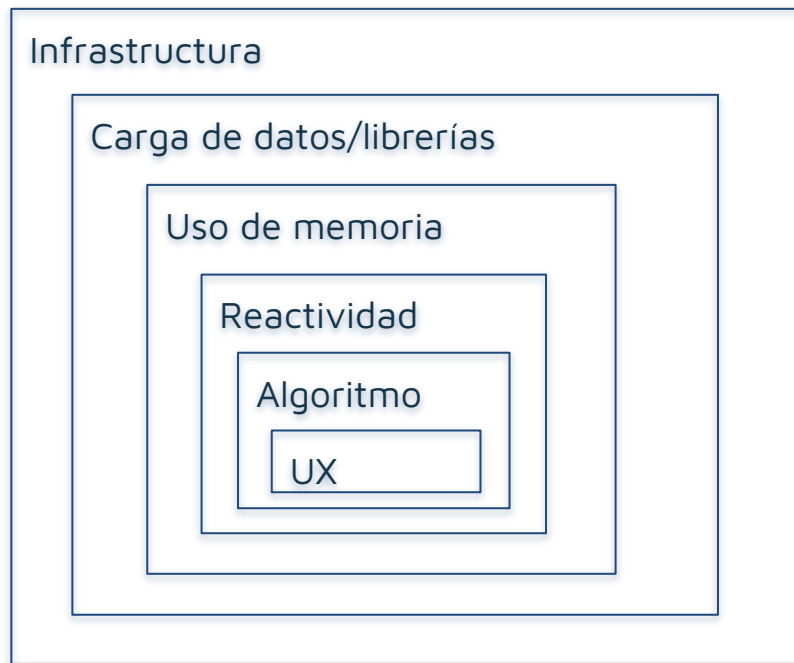
89%

Uso de memoria



Experiencia del usuario

Mejora de la performance: Un campo multidisciplinar



¿Por donde empezar?

Siempre hay lugar para mejorar la performance...

¿Por donde empezar?

Siempre hay lugar para mejorar la performance...



Tiempo del programador



Experiencia del usuario

¿Por donde empezar?

Experiencia del usuario: Escuchar opinión

- ☐ Mejor preguntar antes de optimizar

¿Por donde empezar?

Experiencia del usuario: Escuchar opinión

- Mejor preguntar antes de optimizar

Tiempo del programado:

- Seguir buenas prácticas
- Usar herramientas para identificar cuellos de botella
- Trabajar con ejemplos

Herramientas para el análisis de performance

- `profvis`
- `microbenchmark`
- `lobstr`

Analizar nuestra app con `profvis`

- Identificar cuellos de botella
- Uso de memoria
- Tiempo de cálculo



profile_app.R

Benchmarking para ejemplos/componentes

- Acortar el ciclo de prueba-medir tiempo
- Decisiones específicas en módulos/funciones o componentes



benchmark.R



`ejercicios/benchmarking/ejercicio_benchmarking.R`

Análisis de objetos con lobster

- Uso de memoria
- Estructura
- Conexión con otros objetos

Carga y manejo de datos en Shiny

Segun el tipo de objeto, R dispone de una gran variedad de soluciones adaptadas y optimizadas

Ex: (`igraph`, `Matrix`, ...)

Carga y manejo de datos en Shiny

Segun el tipo de objeto, R dispone de una gran variedad de soluciones específicas y optimizadas

Ex: (`igraph`, `Matrix`, ...)

La estructura más común en una app de Shiny es una tabla de datos, normalmente un `data.frame`

Optimizar el uso de tablas de datos

1. Reducir, seleccionar columnas, filas, guardar resultados
2. Usar paquetes específicos optimizados:
`data.table`, `vroom`, `feather`, `fst`
3. Uso de base de datos
4. Uso de data lake / cloud computing

1 - Cargar solo los datos necesarios

- Preprocesar objetos antes de implementar el app:
- Remover filas, columnas, dividir en multiples objetos, guardar resultados
- Guardar en binario (saveRDS)

2 Usar paquetes específicos ☐

`data.table`

`vroom`

`fst`

`feather`

Cuando usar base de datos

Gran cantidad de datos

Procesar búsquedas en tabla o tablas relacionales

Salvar cambios por el usuario

Compartir datos entre apps

Uso de base de datos en Shiny

dbplyr: Transforma comandos de dplyr en SQL

pool: Manejo de conexiones a servidor Shiny

DBI: Leer, Escribir, operaciones en base de datos

Cuando usar data lake / cloud computing ☐

Diferentes tipos de archivos

Big computing

Conexión con data.lake es más lenta que

Conexión con base de datos, uso de UX

Resumen

- La optimización de performance es un campo multidisciplinar, donde intentamos mejorar el uso de memoria y tiempo de cálculo con el objetivo de una mejor UX
- Antes de optimizar, usar profvis para analizar nuestra app
- Usar sólo los mínimos datos necesarios, especialmente en bucles y funciones
- Benchmarking con ejemplos reducidos
- Conocer optimización de data.frame en R y alternativas en base de datos y cloud computing



`ejercicios/Performance_app/performance_app.R`