

Back to the drawing board

How to quickly mock-up and test your R Shiny application user interface designs



Barbara Mikulášová, Katalyze Data
barbara.mikulasova@katalyzedata.com

Common User Interface Development Issues

- Low user acceptance rate and underutilized application features due to insufficient time spent on designing UI (user interface)
- Delays on the project timeline or fragmented, hard to maintain code base due to inadequate UI prototyping and planning
- The lack of input control to validate user entered information because of poor understanding of users' behaviour on the application
- Lack of clear feature development sequence resulting from poor understanding of the application's core business logic

Avoid these issues by quickly prototyping and testing your UI designs with the following R packages

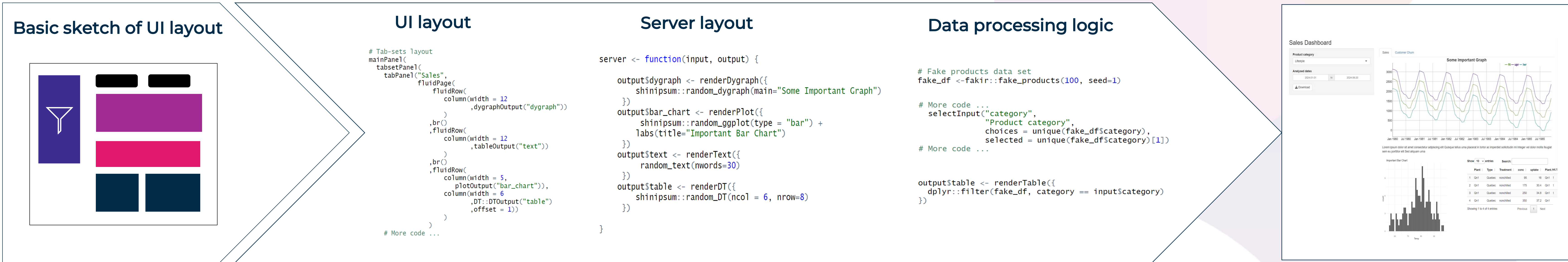
{shinipsum} and {fakir} Wonder Duo

» The solution to your UI development issues !

- {shinipsum}
- Simple random_<element>() functions to generate a variety of UI elements such as fake plots, tables, or statistical output
 - Used as a placeholder for future dashboard elements
 - Compatible with other R and {ggplot2} function calls
- {fakir}
- Simple fake_<type>() functions to generate synthetic datasets with or without data quality issues
 - Used to test and map reactive logic of Shiny elements and test for data quality errors and user inputs related errors



» Shiny application UI layout in 5 minutes !



Accelerate application development process

- The prototyping R packages are cost-effective and a fast way to prototype and test any application
- Developing the UI first helps to finalize the layout of the final application, clarify the key functionalities, and identify potential pain points in the development process in unison with their preventative strategies to ensure timely product delivery
- Fast UI prototype allows A/B testing, to gather targeted feedback on specific design elements, allowing teams to optimise the user experience early in the design process
- Dummy placeholder elements let developers experiment and map out interactions between the dashboard's reactive elements and data
- Identify the best order in which to develop and introduce the application's features. This ensures the development stays on the timeline and the released features do not conflict with the existing code base
- Helps to identify areas of business logic where code modularization of code should be applied for easier scalability and maintenance