Step 1: Capture domain logic

```
server <- function(input, output, session) {

  downloads <- metaReactive({
    cranlogs::cran_downloads(
      input$package,
      from = Sys.Date() - 365,
      to = Sys.Date()
    )
  })

  downloads_rolling <- metaReactive2({
    validate(need(sum(downloads()$count) > 0, "Input a valid package name"))

    metaExpr({
      downloads() %>%
        mutate(count = zoo::rollapply(count, 7, mean, fill = "extend"))
    })
  })

  output$plot <- metaRender(renderPlot, {
    ggplot(downloads_rolling(), aes(date, count)) + geom_line()
  })
}
```

Step 1: Capture domain logic

```
server <- function(input, output, session) {

  downloads <- metaReactive({
    cranlogs::cran_downloads(
      input$package,
      from = Sys.Date() - 365,
      to = Sys.Date()
    )
  })

  downloads_rolling <- metaReactive2({
    validate(need(sum(downloads()$count) > 0, "Input a valid package name"))

    metaExpr({
      downloads() %>%
        mutate(count = zoo::rollapply(count, 7, mean, fill = "extend"))
    })
  })

  output$plot <- metaRender(renderPlot, {
    ggplot(downloads_rolling(), aes(date, count)) + geom_line()
  })
}
```

reactive becomes metaReactive

render functions must be wrapped in metaRender