NHS Low Income Scheme

Kayoung Goffe & Adam Ivison

10/03/2022

The NHSBSA Data Science team recently published an article investigating the take-up of the NHS Low Income Scheme in England. The NHS Low Income Scheme is a service administered by the NHS Business Services Authority, which provides help to people who are not already entitled to help with health costs.

We wanted to try something new to report the findings and tell the story behind the data; and not just create a "usual" dashboard or report. We took inspiration from a scrollytelling article developed by ONS called Exploring local income deprivation. They developed it using Svelte, Layer Cake and Mapbox GL and it inspired our team to try something similar in R, as this is the language our team is familiar with.

If you want to develop a scrollytell report using Javascript, you can use the ONS template. In this blog, we outline how we developed a scrollytell article in `{shiny}`using the `{golem}` framework. We also discuss the challenges we faced to ensure accessibility and how we overcame them by adapting the NHS.UK frontend toolkit.

Golem

As Data Scientists, we look to employ best practice wherever we can. For this complex shiny app, this meant modularising our code by following the `{golem}`framework, in order to package the app as an R package. We achieved this by keeping each scrollytell section separate and packaging up reusable components into functions and modules. Otherwise, it would be a huge monolithic application that would be horrible to debug and maintain.

Our team regularly checks the latest news and R tips on the NHSR community website, Slack, and Twitter and we stumbled across Clinical Development Unit Data Science Team's '{experiencesdashboard}'. The open sourcing of this dashboard was vital in the development of our own product. We used it as a reference guide and also reused some of the ideas in the code.

We have published the 'data-raw' directory (similar to CDU's repository) but not the `data` directory. The `data_raw` directory contains mostly `{dbplyr}` code that was used to transform and aggregate data from a table containing information on NHS Low Income Scheme applicants / applications. The resulting datasets have statistical disclosure control applied to them to prevent accidental disclosure of sensitive information.

Modularised shiny code is much easier to debug and in my opinion, one of the most useful elements of modularised shiny was its reusability across many different modules. We will demonstrate some examples using the NHS frontend toolkit later in the blog.

Scrollytell in R

We used the {scrollytell} R package to develop the article in `{shiny}`.

Below is the code snippet for the scrollytell container inside `app_ui.R`. A scrollytell container is made up of a fixed scrolly graph and multiple scrolly sections.

The public version of this report does not make full use of the fixed scrolly graph. We use it to differentiate the sections when scrolling down the article. However, please keep your eyes peeled as we are planning to open source more code where you can see how this works.

```
scrollytell::scrolly_container(
              outputId = "scrolly",
              scrollytell::scrolly_graph(),
              scrollytell::scrolly_sections(
                scrollytell::scrolly_section(
                  id = "02_what_is_lis",
                  mod_02_what_is_lis_ui("02_what_is_lis_ui_1")
                ),
                scrollytell::scrolly_section(
                  id = "03_who_applies_to_lis",
                  mod_03_who_applies_to_lis_ui("03_who_applies_to_lis_ui_1")
                ),
                scrollytell::scrolly_section(
                  id = "04_applications_over_time",
                  mod_04_applications_over_time_ui("04_applications_over_time_ui_1")
                ),
                scrollytell::scrolly_section(
                  id = "05_what_help_does_lis_provide",
                  mod_05_what_help_does_lis_provide_ui("05_what_help_does_lis_provide_ui_1")
                ),
                scrollytell::scrolly_section(
                  id = "06_take_up_region",
                  mod_06_take_up_region_ui("06_take_up_region_ui_1")
                ),
                scrollytell::scrolly_section(
                  id = "07_take_up_la",
                  mod_07_take_up_la_ui("07_take_up_la_ui_1")
                ),
                scrollytell::scrolly_section(
                  id = "08_spotlight_students",
                  mod_08_spotlight_students_ui("08_spotlight_students_ui_1")
                ),
                scrollytell::scrolly_section(
                  id = "09_final_thoughts",
                  mod_09_final_thoughts_ui("09_final_thoughts_ui_1")
                )
              )
            )
```

Accessibility

Our app needed to meet the NHS accessibility standard. Accessibility testing was new to us and was not an easy task. And it seemed that we had less control over the shiny app.

We were recommended to test using three accessibility tools; WAVE, Lighthouse and Axe Devtools. These are all freely available and easily installed through google extension. The idea was to remove any serious or critical errors from these tests. Some of the errors were easy to fix. For example, we needed to declare page language for accessibility reasons. To fix that issue, we just needed to add one html tag `lang = "en"`.

However, some of the errors were hard to fix. For example, web page content should be able to resize

to 400% without loss of content or functionality. We initially failed the test as some contents overlapped when we zoomed. To solve the more complicated accessibility issues, we used the NHS.UK frontend site. We downloaded the zip file, modified it slightly (see this norem issue), and added it to the `inst/app/www` directory.

NHS.UK frontend toolkit was a great resource as it contained most of the CSS classes we needed. The benefit of using this was that once we applied the proper CSS style our shiny app looked so much like a NHS website! We also referenced this site as well to modify shiny code if necessary. (Examples - NHS.UK frontend design system components). We can show a few examples of how we modified them.

- SelectInput: we wanted to have our selectInput similar to NHS style. We inspected html code from the frontend example page (Select - NHS.UK frontend design system components) and modified the CSS to look like an NHS select Input (rather than default shiny style).

Our shiny code

```
#' selectInput Function
#'
#' @importFrom shiny tagList
nhs_selectInput <- function(inputId,
                            label,
                            choices,
                            selected = NULL,
                            full_width) {

  # Create select input
  nhs_selectInput <- shiny::selectInput(
    inputId = inputId,
    label = label,
    choices = choices,
    selected = selected,
    selectize = FALSE
  )

  # Hack the CSS to look like an NHS select input
  nhs_selectInput$attribs$class <- "nhsuk-form-group"
  nhs_selectInput$children[[1]]$attribs$class <- "nhsuk-label"

  if (full_width) {
    nhs_selectInput$children[[2]]$children[[1]]$attribs$class <- "nhsuk-select form-control"
    nhs_selectInput$children[[2]]$children[[1]]$attribs$style <- "border-radius: 0;"
  } else {
    nhs_selectInput$children[[2]]$children[[1]]$attribs$class <- "nhsuk-select"
  }
  tagList(
    nhs_selectInput
  )
}
```

NHS toolkit select html code

```
<div class="nhsuk-form-group">
  <label class="nhsuk-label" for="select-1">
    Label text goes here
```

```
  </label>
  <select class="nhsuk-select" id="select-1" name="select-1">
    <option value="1">NHS.UK frontend option 1</option>
    <option value="2" selected>NHS.UK frontend option 2</option>
    <option value="3" disabled>NHS.UK frontend option 3</option>
  </select>
</div>
```

Output SelectInput

- Action button for download: we wanted to have our download button similar to NHS style. We inspected html code from the frontend example page (Action Link - NHS.UK frontend design system components) and modified the CSS to look like an NHS action link.

Our shiny code

```
mod_nhs_download_ui <- function(id) {
  ns <- NS(id)
  tagList(
    tags$div(
      style = "position:relative; height:55px;",
      tags$div(
        class = "nhsuk-action-link",
        style = "position:absolute; bottom:0; right:0; margin-bottom:0;",
        shiny::downloadLink(
          outputId = ns("download"),
          class = "nhsuk-action-link__link",
          tags$svg(
            class = "nhsuk-icon nhsuk-icon__arrow-right-circle",
            xmlns = "http://www.w3.org/2000/svg",
            viewBox = "0 0 24 24",
            `aria-hidden` = "true",
            width = "36",
            height = "36",
            tags$path(
              d = "M0 0h24v24H0z",
              fill = "none"
            ),
            tags$path(
              d = "M12 2a10 10 0 0 0 0-9.95 9h11.64L9.74 7.05a1 1 0 0 1 1.41-1.41l5.66
              5.65a11 0 0 1 0 1.42l-5.66 5.65a1 1 0 0 1-1.41 0 1 1 0 0 1
              0-1.41L13.69 13H2.05A10 10 0 0 1 12 2z"
            )
          ),
          tags$span(
            class = "nhsuk-action-link__text",
            "Download Data"
          )
```

```
        )
      )
    )
  )
}
```

NHS toolkit action link html code

```
<div class="nhsuk-action-link">
  <a class="nhsuk-action-link__link"
        href="https://www.nhs.uk/service-search/minor-injuries-unit/locationsearch/551">
    <svg class="nhsuk-icon nhsuk-icon__arrow-right-circle" xmlns="http://www.w3.org/2000/svg"
         viewBox="0 0 24 24" aria-hidden="true" width="36" height="36">
      <path d="M0 0h24v24H0z" fill="none"></path>
      <path d="M12 2a10 10 0 0 0-9.95 9h11.64L9.74 7.05a1 1 0 0 1 1.41-1.41l5.66
      5.65a1 1 0 0 1 0 1.42l-5.66 5.65a1 1 0 0 1-1.41 0 1 1 0 0 1
      0-1.41L13.69 13H2.05A10 10 0 0 1 12 2z"></path>
    </svg>
    <span class="nhsuk-action-link__text">Find a minor injuries unit</span>
  </a>
</div>
```

Output download button

Once we applied the NHS.UK frontend toolkit we successfully passed the accessibility test.

Please take a look at our shiny scrollytell article and check out the code on GitHub. We also created `{golem}`shinyR template.

We were grateful to the NHSR community as it always inspires us to improve our code and we hope we can contribute back to the community by sharing our code. Thank you for taking time to read.

Kayoung Goffe (Data Scientist, NHS Business Services Authority)

Adam Ivison (Data Scientist, NHS Business Services Authority -> Principal Data Scientist, UK Health Security Agency)