Appsilon R consortium

FDA Pilot 4 - Sprint 1 wrap-up

Podman & webAssembly

RConsortium Submissions Working Group

Authors: André Veríssimo, Damian Rodziewicz, Vedha Viyash, Paweł Rzymkiewicz

Period under review: June 19th - July 3rd 2023 *(2 weeks)*

Summary

podman

- Deploys Pilot2 shiny app locally without installing R or packages
- Flexible configuration
- Requires podman & access to base image *(docker.io)*

webR *(webAssembly)*

- Learning sprint to understand current status
- Working version of a teal minimal application
- 22 dependencies from Pilot2 not easily ported
  - Most are golem-based

Appsilon

Agenda

1. Podman
2. webR *(webAssembly)*
3. Lessons learnt
4. Future work
5. Open discussions & Impact

Appsilon

podman

[Appsilon/experimental-fda-submission-4-podman](https://github.com/Appsilon/experimental-fda-submission-4-podman)

Goal: Container-based method to deploy Pilot 2 Shiny App

What we did:

- Configurable Podman Dockerfile / docker-compose.yml
  - R version
  - Registry / organization name / image name *(differences between docker.io and ghcr.io)*
- Documentation on creating the container
- CI: Automated build on amd64 and arm64 platforms

podman short demo

```r
ARG R_VERSION=4.2.0
ARG IMAGE_REGISTRY=docker.io
ARG IMAGE_ORG=rocker

FROM $IMAGE_REGISTRY/$IMAGE_ORG/r-ver:$R_VERSION

LABEL org.opencontainers.image.licenses="GPL-3.0-or-later" \
      org.opencontainers.image.source="https://github.com/Appsilon/experimental-fda-submission-4-podman" \
      org.opencontainers.image.vendor="Appsilon" \
      org.opencontainers.image.authors="André Veríssimo <andre.verissimo@appsilon.com>, Vedha Viyash <vedha@appsilon.com>"

RUN apt-get update --quiet \
  && apt-get install -y --quiet \
    curl \
    libssl-dev \
    libcurl4-openssl-dev \
    libxml2-dev \
    libfontconfig1-dev \
    libharfbuzz-dev libfribidi-dev \
    libfreetype6-dev libpng-dev libtiff5-dev libjpeg-dev \
  && apt-get autoremove -y --quiet \
  && apt-get clean --quiet \
  && rm -rf /var/lib/apt/lists/*

ENV RENV_PATHS_ROOT=/renv_cache

COPY ./renv_cache/ $RENV_PATHS_ROOT

ARG LOCAL_DIR=./submissions-pilot2
ARG APP_DIR=/usr/local/src/submissions-pilot2

COPY $LOCAL_DIR $APP_DIR

WORKDIR $APP_DIR

# Prevents RENV from mistakenly download from teal.* remotes (as the dependencies are
#  already defined in renv.lock).
RUN Rscript \
  -e "options(\"renv.config.install.remotes\" = FALSE)" \
  -e "renv::restore()"

ARG R_SCRIPT=./entrypoint.R

COPY $R_SCRIPT $APP_DIR/entrypoint.R

CMD ["Rscript", "entrypoint.R"]
```

(build arguments for flexibility)

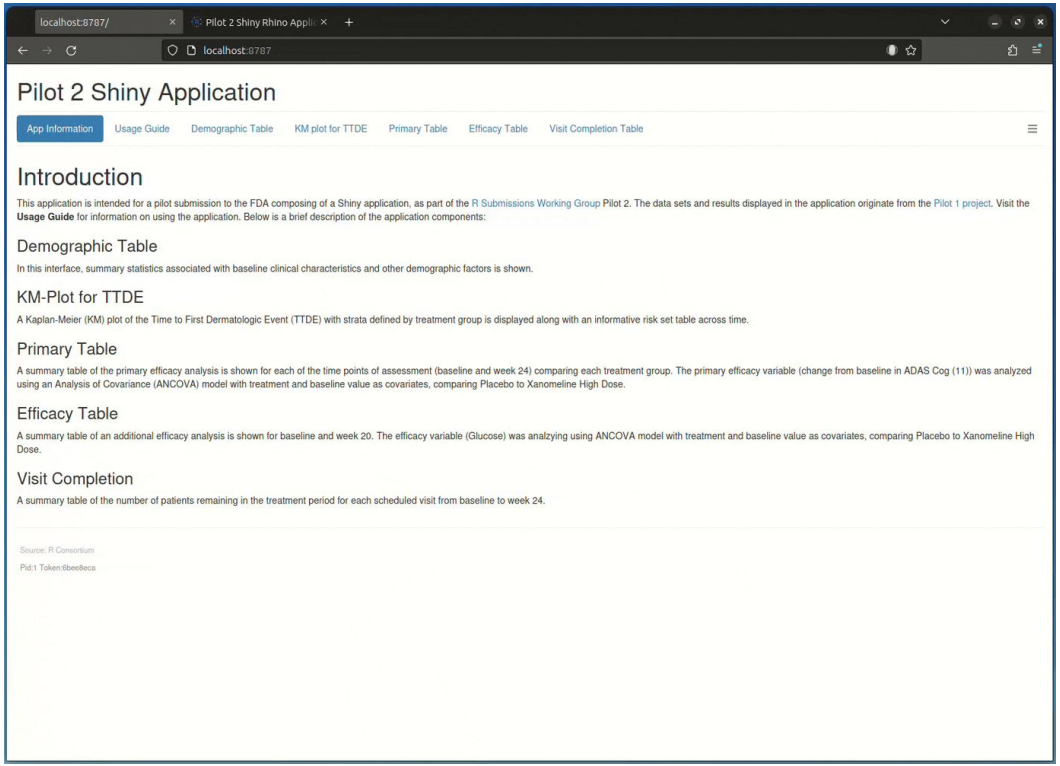
1. Install system requirements

(use host renv cache)

(configurable shiny directory)

2. Install R packages

3. Init script



(fallback to video)

webR *(webAssembly)*

Goal: Explore webR potential as a method to deploy a Shiny application on the browser without additional requirements *(setup R / podman / ~~a server~~)*

What we did:

- Understand the build process
- Manually build and curate 34 unsupported R packages
  - Focus on supporting teal → Pilot2 dependencies
- Discussion with  George Stagg *(principal webR developer @Posit)*
  - Very interested in the pilot
  - One of the outputs is the standalone shiny webR template for the demo
- 1 upstream bug fix
  - 1 more pending *(allows for `{teal.*}` build while not on CRAN)*

Resources

r-wasm/**webr**

- Build from source
- Get from npm
- Get from CDN

r-wasm/**webr-repo**

- Build R packages for webR

georgestagg/shinylive

- Experimental R ShinyLive with webR

georgestagg/shiny-standalone-webr-demo

- Demo of standalone webR shiny app

fork: Appsilon/experimental-teal-webr-demo

webR docs

beware: unstable

webR limitations

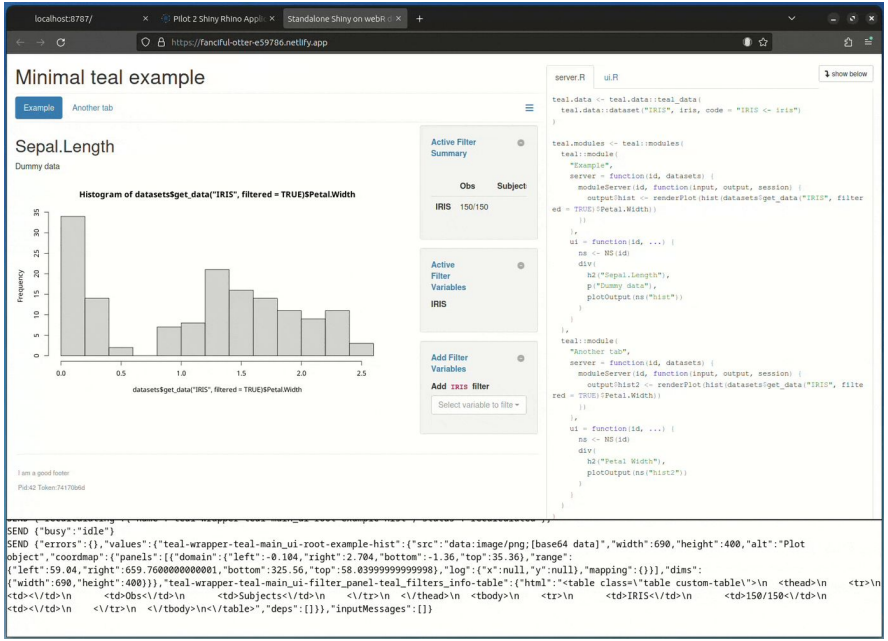
some of the current limitations

- Very low coverage of CRAN packages
  - Non-supported R package need to be build manually and deployed
- No testing framework in place
  - Currently it's not possible to validate packages
  - {testthat} package is available, but it's not usable to test package
- Slow loading
  - Packages are being downloaded and installed in runtime
  - Alternative
- Unstable / incomplete API and documentation
  - No documentation on package building
  - Changes very fast and without warning
    - This is at a pre-alpha stage
- Some packages only have limited functionality despite being available
  - *{openssl}* for example

webR short demo

live demo: https://**teal-webr-example.netlify.app** *(it might take a long while to fully load)*

Appsilon/experimental-teal-webr-demo *(fork this & deploy yourself on netlify)*



Custom teal module just as Pilot 2

Some stats:

- **34**/100 packages built during this sprint
- `{vctrs}` & `{styler}` required patches

Upstream *r-wasm/webr-repo*

- 1 upstream bug fix to build R packages
- 1 pending PR on using custom repository to allow for non-cran packages (`{teal}`, `{teal.*}`)

(fallback to video)

Appsilon

Lessons learnt

- Podman is a valid alternative to Docker
  - Speeds up development to have volume support during `build`
- webR has a good potential, but it needs to mature
  - Update on state of the project
  - Deeper understanding of current webR internals
  - Poor support for development in arm64 environment

Appsilon

Future work

technical improvements

Podman:: Some technical improvements

- **Publish container image to container registry**

- **Understand FDA access to registries**
  - Include image as part of submission?

webR:: Continue exploratory work

- Cleanup of Pilot2 to **remove unnecessary dependencies** *(devtools, golem, XML2, ...)*

- Podman/Docker image to help build webR & packages
  - Convoluted process that still requires **a lot of manual fixes and iterations**

Open discussions & Impact

What is needed to take this project further?

How can Appsilon support the overall impact of the project?

Appsilon R consortium

Thank you

localhost:8787/ | Pilot 2 Shiny Rhino Appli

localhost:8787

Pilot 2 Shiny Application

App Information | Usage Guide | Demographic Table | KM plot for TTDE | Primary Table | Efficacy Table | Visit Completion Table

Introduction

This application is intended for a pilot submission to the FDA composing of a Shiny application, as part of the R Submissions Working Group Pilot 2. The data sets and results displayed in the application originate from the Pilot 1 project. Visit the **Usage Guide** for information on using the application. Below is a brief description of the application components:

Demographic Table

In this interface, summary statistics associated with baseline clinical characteristics and other demographic factors is shown.

KM-Plot for TTDE

A Kaplan-Meier (KM) plot of the Time to First Dermatologic Event (TTDE) with strata defined by treatment group is displayed along with an informative risk set table across time.

Primary Table

A summary table of the primary efficacy analysis is shown for each of the time points of assessment (baseline and week 24) comparing each treatment group. The primary efficacy variable (change from baseline in ADAS Cog (11)) was analyzed using an Analysis of Covariance (ANCOVA) model with treatment and baseline value as covariates, comparing Placebo to Xanomeline High Dose.

Efficacy Table

A summary table of an additional efficacy analysis is shown for baseline and week 20. The efficacy variable (Glucose) was analzying using ANCOVA model with treatment and baseline value as covariates, comparing Placebo to Xanomeline High Dose.

Visit Completion

A summary table of the number of patients remaining in the treatment period for each scheduled visit from baseline to week 24.

Source: R Consortium

Pid:1 Token:tibeelteca

(click here to go back to presentation)

localhost:8787/ | Pilot 2 Shiny Rhino Appli | Standalone Shiny on webR

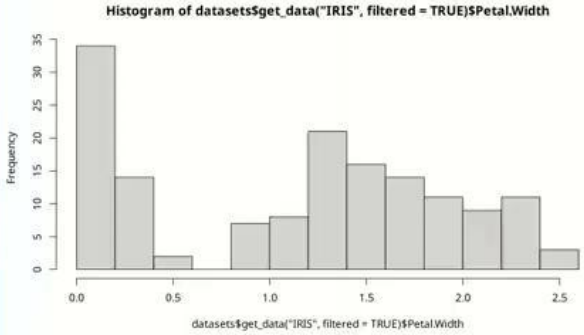
https://fanciful-otter-e59786.netlify.app

Minimal teal example

Example | Another tab

Sepal.Length

Dummy data



Active Filter Summary

	Obs	Subject
IRIS	150/150	

Active Filter Variables

IRIS

Add Filter Variables

Add IRIS filter

Select variable to filte

I am a good footer

Pid:42 Token:74170b6d

server.R | ui.R | show below

```
teal.data <- teal.data::teal_data(
  teal.data::dataset("IRIS", iris, code = "IRIS <- iris")
)

teal.modules <- teal::modules(
  teal::module(
    "Example",
    server = function(id, datasets) {
      moduleServer(id, function(input, output, session) {
        output$hist <- renderPlot(hist(datasets$get_data("IRIS", filter
ed = TRUE)$Petal.Width))
      })
    },
    ui = function(id, ...) {
      ns <- NS(id)
      div(
        h2("Sepal.Length"),
        p("Dummy data"),
        plotOutput(ns("hist"))
      )
    }
  ),
  teal::module(
    "Another tab",
    server = function(id, datasets) {
      moduleServer(id, function(input, output, session) {
        output$hist2 <- renderPlot(hist(datasets$get_data("IRIS", filte
red = TRUE)$Petal.Width))
      })
    },
    ui = function(id, ...) {
      ns <- NS(id)
      div(
        h2("Petal Width"),
        plotOutput(ns("hist2"))
      )
    }
  )
)
```


```
SEND {"busy":"idle"}
SEND {"errors":{},"values":{"teal-wrapper-teal-main_ui-root-example-hist":{"src":"data:image/png;[base64 data]","width":690,"height":400,"alt":"Plot
object","coordmap":{"panels":[{"domain":{"left":-0.104,"right":2.704,"bottom":-1.36,"top":35.36},"range":
{"left":59.04,"right":659.7600000000001,"bottom":325.56,"top":58.03999999999998},"log":{"x":null,"y":null},"mapping":{}}],"dims":
{"width":690,"height":400}}},"teal-wrapper-teal-main_ui-filter_panel-teal_filters_info-table":{"html":"<table class=\"table custom-table\">\n  <thead>\n    <tr>\n
<td><\/td>\n      <td>Obs<\/td>\n      <td>Subjects<\/td>\n    <\/tr>\n  <\/thead>\n  <tbody>\n    <tr>\n      <td>IRIS<\/td>\n      <td>150/150<\/td>\n
<td><\/td>\n    <\/tr>\n  <\/tbody>\n<\/table>","deps":[]}},"inputMessages":[]}
```


(click here to go back to presentation)

Resources

r-wasm/**webr**

- Build from source
- Get from npm
- Get from CDN

r-wasm/**webr-repo**

- Build R packages for webR

georgestagg/shinylive

- Experimental R ShinyLive with webR

georgestagg/shiny-standalone-webr-demo

- Demo of standalone webR shiny app

fork: Appsilon/experimental-teal-webr-demo

webR docs

🤖 beware: unstable

Resources

r-wasm/webr

Get from CDN

server.R

```
library(httpuv)
runServer(host = "127.0.0.1", port = 8080,
  app = list(
    staticPaths = list(
      "/" = staticPath(
        ".",
        headers = list(
          "Cross-Origin-Opener-Policy" = "same-origin",
          "Cross-Origin-Embedder-Policy" = "require-corp"
        )
      )
    )
  )
)
```

index.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Using WebR</title>
    <script>
      function displayMessage(message) {
        document.body.innerHTML = message;
      }
    </script>
  </head>
  <body>
    <script type="module">
      displayMessage("Please wait while the WebR is initiated.");
      import("https://webr.r-wasm.org/latest/webr.mjs").then(
        async ({ WebR }) => {
          const webR = new WebR();
          await webR.init();
          displayMessage("WebR is Initiated!");
        }
      );
    </script>
  </body>
</html>
```

```
Rscript server.R
```

Run the server and check the browser:
http://127.0.0.1:8080/

Resources

r-wasm/webr

Get from npm

```
npm i @r-wasm/webr
```

```
import { WebR } from '@r-wasm/webr';
const webR = new WebR()
await webR.init();
```

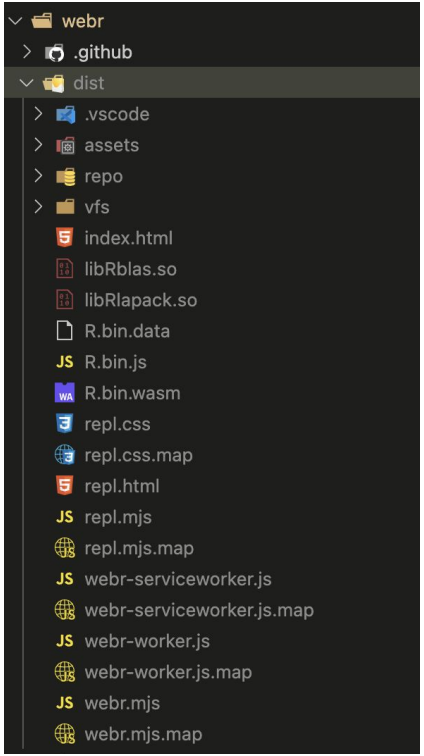
Resources

r-wasm/webr

Build from source

1. ~/webr > ./configure
2. ~/webr > make

Builds the webr in the /dist



Resources

r-wasm/webr-repo

Build R packages for webR

Note: Make sure that webR is built from source on your machine

1. ~/webr-repo > make
2. ~/webr-repo > make pkg-<PACKAGE_NAME>

Updates the R packages and
Builds the specified R package

Output of this would be the /repo would be populated with the R package

Resources

r-wasm/webr

Create npm package

~/webr/src > make package

Creates **r-wasm-webr-0.1.2-dev.tgz**

Note: In order to include the built R packages in the webr npm package include the webr-repo libs in the make config file called **~webr/src/.webr-config.mk** like this: **WEBR_LIB=/r-wasm/webr-repo/lib**

Resources

georgestagg/shinylive

Shiny Live

Experimental R ShinyLive with webR

Appsilon

Resources

Shiny Demo Demo of standalone webR shiny app

georgestagg/shiny-standalone-webr-demo

fork:
Appsilon/experimental-teal-webr-demo