PRO TIPS

from RStudio Customer Success

RStudio

Creating Efficient Workflows with `pins`

When Workflows Get Clunky

- Does your workflow contain `read.csv()` for a data file that your colleague emails to you after every update?
- Are you working with ephemeral data that refreshes frequently?
- Do you need to redeploy your app every time the supporting data is updated?
- Do multiple apps you've created need to call a model that you developed?
- Are you saving `.R` or `.RData` objects to be called later?

The habits and methods (??) above can thrust a speedbump in your workflow. Often data scientists struggle to share and reuse content; knowing how and where to store resources, especially resources that can't be put into a database, is a persistent problem.

This is where Pins come in. RStudio developed Pins to make the process of discovering, caching, and sharing resources simpler.

If you're asking colleagues to download files before running your code, redeploying an entire app just to update the model or data in it, or if many pieces of your content are pointed to the same dataset, read on.

What are Pins?

Pins are almost exactly what they sound like: a way to store and then remotely access R and Python objects. Pins live on boards. A board can exist on a number of services(? right word?), including RStudio Connect, S3, Google Cloud - you can even integrate them into your website.

Once resources are pinned, you and your colleagues can also discover or find them. Pins on sites like Kaggle let you search well-known data repositories that are guaranteed to contain datasets, instead of searching on the internet and hoping to find what you need. And when you use pins internal to your organization on RStudio Connect, you enhance the discoverability of your datasets with the same access controls you're accustomed to for any content published to Connect.

How to Use Pins / Example or walk me through my first pin

<structure this like a recipe. what ingredients do you need?> ingredients - data set tools - API key, connect (you can do this on your own, aren't reliant on someone else) There are four main steps to using Pins. First, install the package, get your R session to connect with your board, and then Pin a resource. Finally, you can get resources. Let's walk through each step using RStudio Connect as an example.

Before you can publish your first pin to the RStudio Connect board, you have to get an API key from Connect and save this to your system environment variables[1]. This is not as intimidating as it sounds - we'll walk you through the steps below!

1. Generate an API key in RStudio Connect.[2] Give this key any name you like and be certain to copy the value to your clipboard.
2. Save this API key to your system environment variables - Switch back to your RStudio IDE and run the following in the Console, pasting your API key value into the command:

```
Sys.setenv("RSC_API_KEY" = "key value")
```

3. "Register" the board.[3] Add the following to your script, inserting your own server address:

```
pins::board_register(
    "rsconnect",
    server = "https://rstudioconnect.example.com/",
```

[1] This is the best practice rather than hard coding the API key into your script. Remember, your keys should be kept secure!

[2] for full instructions, see https://docs.rstudio.com/connect/user/api-keys/

[3] Don't let the vocabulary throw you. This just means you're identifying a location where you can store resources; think of it as registering (or recognizing) the board location within your current session.

```
    key = Sys.getenv("RSC_API_KEY")
    )
```

4. Now you're going to Pin your first resource. Put your object on the rsconnect board with:

```
pin(my_data, description = "Super Cool data set", board = "rsconnect")
```

Code to retrieve the pin will now be provided on the deployed content in RStudio Connect: `setsData <- pin_get("username/my_data", board = "rsconnect")`

You're ready to publish your Pin. The first time you publish, it will fail saying "[Connect] Message: 'Invalid API key, the API key is empty.' " because there is no key on the environment variable on connect. Go put RSC_API_KEY in environment variables and refresh.

How to share your pin with others

Retrieving a pin

Where do i go for help / other resources / are there any other considerations?

- current release version as of time of writing, where to get the package

Should we add this in somehow? Go try on your own: Pins is particularly useful for sharing R objects when the objects are

Relatively small (a few hundred megabytes at most) Reused across multiple pieces of content Only needed in their most current form