5 Projetos e Pastas

Quando iniciando um novo projeto de pesquisa, temos as boas intenções de organizar todos nossos arquivos para não sofrer mais tarde (como amanhã!) quando voltamos para nossos dados. Nestes dias de hoje da ênfase em replicabilidade em ciência, organização dos projetos fica ainda mais importante. Cada artigo, estudo, poster, boa ideia, ele precisa seu próprio espaço no seu computador para não ser confuso com outros projetos. Geralmente, cientistas aceitam essa ideia.

Colocar ele em prática é outra coisa. Por isso, precisamos pensar um pouco em nosso fluxo de trabalho (*workflow*). Recebemos arquivos dos sequenciadores, das outras maquinas, dos arquivos que criamos com dados determinados de mão, dados que vem dos arquivos oficiais (e.g., demográficos), e de vários outros fontes. Temos uma pasta chamada `MEU_PROJETO` no algum nível do seu disco rígido. Pense bem: o que você vai fazer com todos os arquivos que ficarão dentro desta pasta. Tem uma organização deles ou são jogadas em um grupo. Eu sei que um projeto do tamanho médio pode ter até 500 arquivos. Vai lembrar onde todos são e o que quer dizer os nomes de todos daqui três meses.

Organização de um projeto é uma tarefa que precisa um pouco de pensamento. Outro fator para lembrar. Numa época em que dividimos a responsabilidade para as pesquisas com vários colegas, temos que fazer possível que nossos programas/scripts funcionam para todos os membros da equipe.

Falamos mais cedo das *working directories*, as pastas ativas. A pasta ativa fica no final de cadeia das pastas que iniciam com a pasta *root* ou raiz. Você pode estabelecer que a pasta ativa é qualquer pasta que você quer com o comando `setwd()`. Por exemplo, eu posso trabalhar na pasta deste curso com o comando seguinte:

```
setwd("/Users/jameshunter/Documents/MAD/MAD-Biomedicina-2019")
```

Mas, este cria um grande problema. Ninguém fora de mim em nosso grupo tem essa

pasta no seu computador. Aposto que eu seja o único com um usador com o nome de "jameshunter". Para qualquer outro computador ou cópia deste arquivo, o comando vai retornar um erro. A solução primitiva para essa problema é de copiar e colar a pasta certa em todos os lugares onde este aparece nos programas. Hiper-chato!

5.1 Projetos em RStudio

RStudio oferece um sistema para você tratar de todos seus projetos diferentes com nomes e espaços distintos. É um dos aspetos mais poderosos de RStudio. Um projeto de RStudio guarda todos os arquivos de um projeto juntos mas separados de todos os outros arquivos de seu computador numa pasta designado para o projeto em que você está trabalhando. Quando você estabelece um novo projeto ou trocar de um para um outro, RStudio inicia uma nova instância do programa para este projeto sempre seja independente dos outros. Você pode ter múltiplas projetos abertos no mesmo tempo apesar se seu computador tem memoria limitada (< 4GB) é provavelmente não uma boa ideia de ter mais que um projeto rodando simultaneamente.

Um dos pontos mais importantes sobre projetos é que RStudio vai usar a pasta onde você cria o projeto como a pasta ativa que fará muito para aliviar as ansiedades que vêm dos conflitos de pastas. Alguns professores acham que este ponto é tão importante que uma delas (Profa. Jenny Bryan da Universidade de British Columbia e de RStudio), uma autoridade sobre uso de R, disse numa presentação

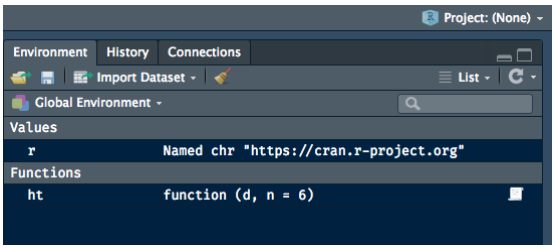
> If the first line of your R script is
>
> ```
> setwd("C:\Users\jenny\path\that\only\I\have")
> ```
>
> I will come into your office and SET YOUR COMPUTER ON FIRE

Como resultado, se você não quer que Profa. Bryan voa de Vancouver para São Paulo para ela incendiar seu computador, é melhor usar a habilidade de RStudio de criar projetos para seus trabalhos.

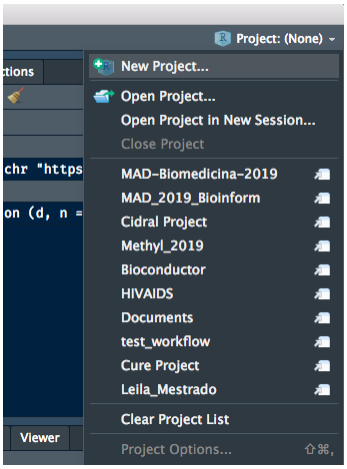
VSS–Sempre crie um novo RStudio projeto para cada projeto

5.2 Criar Um Novo Projeto

Na direta da tela acima das abas de “Environment, History, …”, você vai ver um logotipo de R com o texto “Project:(None)” mais uma seta que pode clicar.

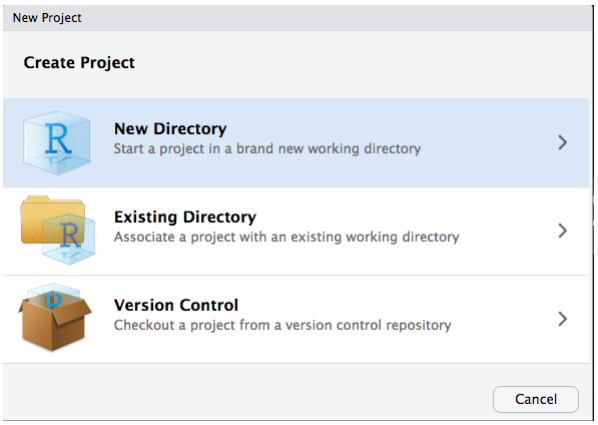


Esse texto indica que ainda não tem um projeto ativo. Podemos iniciar a criação de um projeto clicando nessa seta. No menu drop-down, podemos ver a opção de criar um novo projeto (*New Project*) ou selecionar um dos projetos com que você já tinha trabalhado.



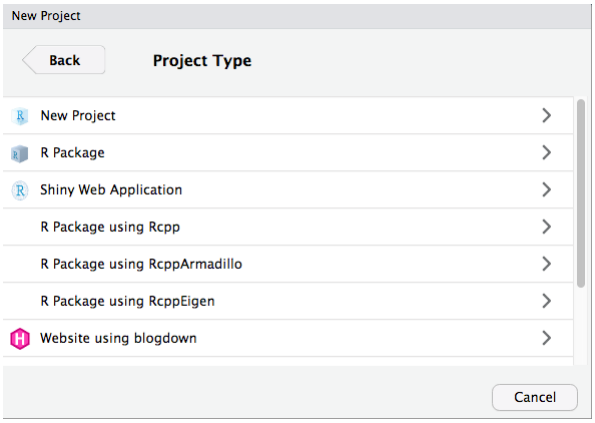
(#fig:proj_2)Project Menu

Queremos selecionar *New Project*, que abrirá a janela que permite que você selecionar onde quer localizar o projeto no seu disco. Você pode escolher de criar uma nova pasta (*directory*) para seu projeto ou colocar ele em uma pasta existente. Se você já tem um espaço no seu computador para seu projeto atual, pode escolher ele com essa segunda opção.



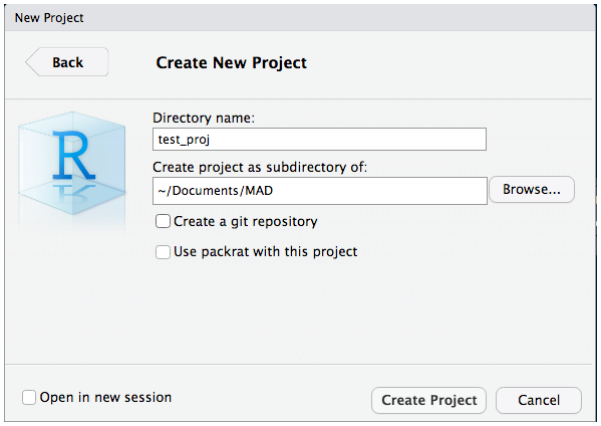
(#fig:proj_3)Create Project

Mas, neste caso, vamos criar uma nova pasta. Então, clique na área de *New Directory* e a janela seguinte aparecerá. Existem vários tipos de projetos que pode fazer em RStudio. Mas, vamos selecionar o primeiro, o projeto basicão, *New Project*.



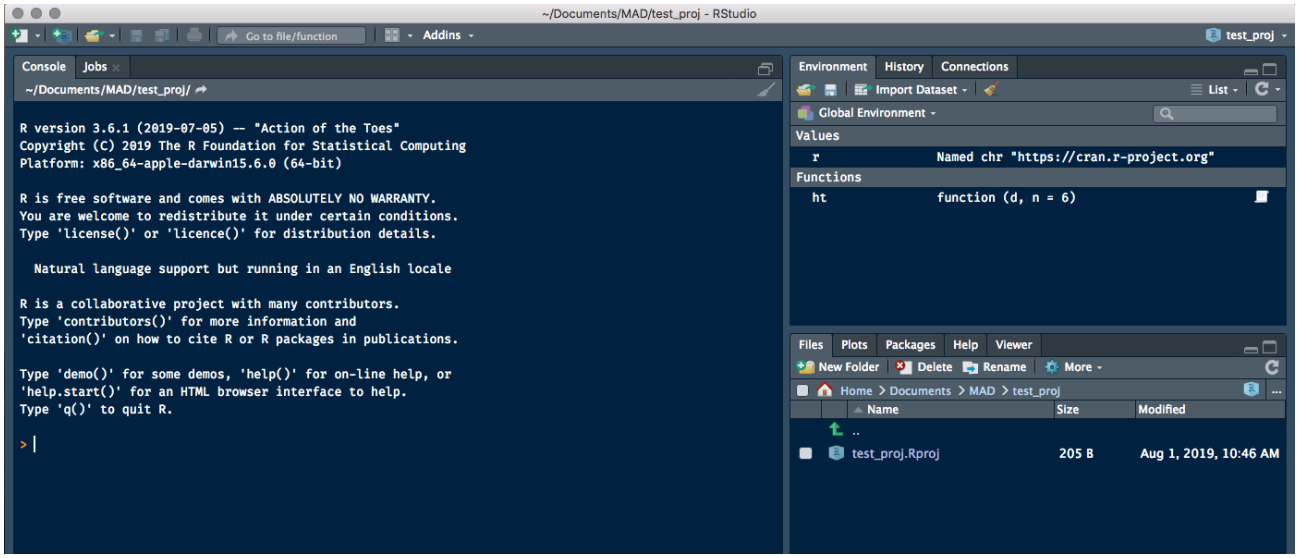
(#fig:proj_4)Tipo de Projeto

Essa escolha levará você para a tela seguinte em que você pode dar um nome para a pasta que conterá o projeto.



(#fig:proj_5)Projeto Novo Nome

Aqui, dei o nome “test_proj” ao novo projeto. Quando você clique no botão `Create Project` , o sistema criará uma pasta com este nome. Você pode ver que este pasta vai ficar como uma sub-pasta da pasta `~/Documents/MAD` que é apropriado para meu computador. Você pode colocar ela num local conveniente para você. O til `~` ao início da sub-pasta quer dizer a pasta raiz do computador, como "C:" em Windows. Pode ver aqui também dois caixas para selecionar controle de versão *git* ou *packrat*. Por enquanto, deixa-elas vazias. São capacidades de R e RStudio mais avançadas. Para finalizar a criação do projeto, clique no botão `Create Project` .



(#fig:proj_6)Novo Projeto Completo

Agora, no parte da cima a direta, você vê a indicação que você está trabalhando com o projeto `test_proj` . Também, na margem do Console, pode ver o nome da pasta ativa. Também fica em azul a cima da lista dos arquivos. Finalmente, RStudio criou um arquivo `test_proj.Rproj` que guarda as informações para avisar o programa como abrir este projeto no futuro.

5.2.1 SubPastas

Por costume e para ajudar na organização do projeto que pode durar meses ou anos, crio também subpastas para conter documentos de tipos diferentes, como scripts, textos, gráficos, etc. A estrutura que geralmente uso para projetos de pesquisa é o seguinte:

- projeto
  - data_munge
  - data_raw
  - docs
  - graficos
  - outros
  - scripts_analise
  - scripts_limpar
  - scripts_misc
  - slides

5.2.1.1 Algumas Notas sobre as SubPastas

1. Separar os dados. Deixa os dados originais que você importou dos sequenciadores, outras maquinas ou mesmo das planilhas de Excel na pasta `data_raw` .

VSS Esses arquivos são os originais. Se você muda eles, eles seriam corruptos.

Salve versões que você modifica na pasta `data_munge` . Fazer toda a limpeza, restruturação, etc. nessas versões ali, não nos originais. Se você esquecem essa regra, não seria Jenny Bryan que vem incendiar seu computador. Eu ou seu orientador faria! (Piada, mas piada séria!)

2. Divido scripts (programas) baseado em tipos. Os que uso para limpar dados, frequentemente uso versões semelhantes em vários projetos. Então, é simplesmente uma conveniência, mas separo eles dos outros que são mais particular para projeto individual. Também, gosto de saber onde são os slides e os documentos que criei (ou em RMarkdown ou em PowerPoint).

Lembre que um projeto importante (tese ou dissertação, por exemplo), pode ter um número grande de arquivos (centenas ou milhares). Criar uma estrutura para guardar eles ajudará sua sanidade quando o dia de apresentação do trabalho aproxima.

5.3 Onde Estamos, Professor? "*We're HERE*"

Criar um projeto é o primeiro passo para evitar a bagunça que resulta de ter vários projetos com várias pastas e vários colegas trabalhando nos seus próprios computadores. Por exemplo, meu *file path* da raiz para pasta de projeto no meu Mac Air pode ser:

- "/Users/jameshunter/Documents/MAD/test_proj"

Mas, o *file path* para meu colega Marcelo pode ter o texto no seu computador de Windows pode ser:

- "c:_proj"

Muito diferente. Se eu tinha colocado uma referência à subpasta que contem todo meu *file path* numa linha de código, não funcionará no computador de Marcelo. Precisamos usar uma segunda função da R para evitar esse problema.

Um dos pacotes que você tem na sua instalação de R agora é chamada `here`. Este faz uma coisa importante, mas faz ele com elegância. Ele avisa o que é o *file path* inteiro para chegar a pasta ativa (*working directory*).

Por exemplo, podemos ver a pasta ativa deste documento e toda a cadeia que liga ele ao raiz com a função `here()`.

```
here::here()
```

```
## [1] "/Users/jameshunter/Documents/MAD/MAD-Biomedicina-2019"
```

Para mostrar os gráficos neste capitulo e outros, eu guardo todos os arquivos de gráfico numa subpasta de `MAD-Biomedicina-2019` chamado gráficos. Para mostrar nesses capítulos os gráficos, eu posso usar o comando primeiro para especificar onde o sistema pode achar o subpasta `graficos`.

```
gr <- here::here("graficos")
gr
```

```
## [1] "/Users/jameshunter/Documents/MAD/MAD-Biomedicina-2019/graficos"
```

Agora, posso usar a variável `gr` em qualquer computador para apontar para subpasta `graficos` porque `here()` mostrará a subpasta `graficos` para aquele computador. A sequência de dois dois-pontos no comando separando o pacote `here` do comando `here` é a maneira de usar uma função de um pacote sem precisar carregar todo o pacote com `library()` ou `librarian::shelf()`.