**How to build R package**

**Use of packages in R**

1. In RStudio, packages are a way of organizing and sharing R code, functions, and data. Packages are essentially collections of R code and documentation that can be easily installed and loaded into an R session.

2. There are several reasons why packages are useful in RStudio:

3. Modularity: Packages allow you to modularize your code into smaller, more manageable pieces. This makes it easier to maintain and update your code over time.

4. Reusability: By creating a package, you can make your code and functions available for others to use. This promotes code reuse and collaboration.

5. Versioning: Packages are versioned, which means that you can track changes to your code over time and ensure that users are always working with the most up-to-date version.

6. Dependency management: Packages can depend on other packages, which means that they can automatically install and load the necessary dependencies when they are installed.

7. Documentation: Packages can include documentation for functions, datasets, and other objects. This makes it easier for users to understand and use your code.

In summary, packages are a powerful tool for organizing and sharing R code in RStudio, and they offer several benefits such as modularity, reusability, versioning, dependency management, and documentation.

**Here are the steps to build R package**

1. Open RStudio and click on File -> New Project.

2. In the New Project dialog box, select "New Directory" and then "R Package".

3. Choose a name for your package and select a directory to save it in.

4. Rtudio will automatically generate some files for your package, including a DESCRIPTION file, an R folder, and a man folder.

5. In the R folder, create an R script file with the code for your package.

6. In the man folder, create a separate . Rd file for each function in your package. These files should contain documentation for your functions.

7. Update the DESCRIPTION file with information about your package, including the title, description, author, and any required packages.

8. Build and check your package by clicking on "Build" and then "Check" in the "Build" tab in RStudio.

9. If there are any errors or warnings, fix them and rebuild your package until it passes all checks.

10. Once your package is ready, you can submit it to a repository such as CRAN or GitHub for others to use.

11. Decide on a package name and create a new package directory: Choose a unique and descriptive name for your package and create a new directory with that name. This directory will serve as the top-level directory for your package.

12. Add necessary files and folders: Within your package directory, you'll need to create several files and folders that are required for an R package. These include: a DESCRIPTION file that contains metadata about your package, a NAMESPACE file that controls the visibility of your package's functions, a man folder that contains documentation files for your package's functions, and a R folder that contains the actual R code for your package.

13. Write package code: Write R code that defines the functions and objects included in package.

14. Add documentation: Create documentation files for each of the functions in your package. These files should provide a description of the function, its arguments, and its return value.

15. Build and check the package: Use the devtools package to build and check your package. This will create a source package and run a series of tests to ensure that your package is working as expected.

16. Submit your package to a repository: If you want to make your package publicly available, you can submit it to a repository such as CRAN (Comprehensive R Archive Network) or GitHub.

17. There are many tutorials and resources available online that provide more detailed guidance on each of these steps. You can also use the devtools package to automate many aspects of the package creation process.

Once the package is ready, we can submit it to a repository such as CRAN or GitHub for others to use.

**Bold fields are mandatory**:

1. Package: This field specifies the name of your package and is required. It should be bolded.

2. Title: This field provides a brief description of your package and is also required. It should be bolded.

3. Description: This field provides a more detailed description of your package and what it does. It is required and should be bolded.

4. Author(s): This field lists the author or authors of the package. It is required and should be bolded.

5. Maintainer: This field specifies the person or organization responsible for maintaining the package. It is required and should be bolded.

6. License: This field specifies the license under which the package is distributed. It is required and should be bolded.

7. Imports: This field lists the names of other packages that your package depends on. It is an important field and is often bolded.

8. Suggests: This field lists the names of other packages that are suggested but not required for your package to work. It is an important field and is often bolded.

9. By bolding these fields in DESCRIPTION file, we can help make it clear which fields are required, important, or have a specific purpose in the context of your R package.

HERE is my own package

Package: mynewpackage

Type: Package

Title: catsdata

Version: 0.1.0

Authors@R: jhansi chengala

Maintainer:" jhansi@rstudio.com "

Description: A small R package that contains a dataset of 50 cat breeds with corresponding information on their average weight, lifespan, and temperament. The package also includes a function for calculating the median weight of a given breed, making it easy to explore and analyze feline data in R.

License: CC0

Encoding: UTF-8

LazyData: true