**The Shiny ecosystem:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **N°** | **Thematic - usage** | **Package** | **Created by** | **Description** |
| 1 | Create web apps | shiny | RStudio | RStudio’s Framework for creating web apps in R |
| 2 | Create web dashboards | shinydashboard | RStudio | RStudio’s Framework for creating web dashboards in R |
| 3 | Create beautiful web dashboards | [shinydashboardPlus](https://www.rdocumentation.org/packages/shinydashboardPlus/versions/0.7.0) | RinteRface | RinteRface’s Framework for creating enhanced (beautiful) web dashboards in R |
| 4 | Widgets HTML | shinyWidgets |  |  |
| 5 | UI – Add UI materials | [shinymaterial](https://ericrayanderson.github.io/shinymaterial/) |  | Implements Material Design in Shiny Applications |
| 6 | Add JS chunks to shiny | [shinyalert](https://github.com/daattali/shinyalert#examples) | Dean Attali | Easily create pretty popup messages (modals) in Shiny. Modals can contain text, images, OK/Cancel buttons, an input to get a response from the user, and many more customizable options.  Uses the [**sweetalert**](https://github.com/t4t5/sweetalert) JavaScript library to create simple and elegant popups (modals) in Shiny |
| 7 | USE JS in Shiny | [shinyjs](https://deanattali.com/shinyjs/) | Dean Attali | A simple Shiny module for putting swipe based interfaces into shiny apps. Ever felt like Shiny wasn't "tinder-y" enough? Well here you go. |
| 8 | Create web apps & sites | [ArgonR](https://github.com/RinteRface/argonR) | RinteRface | ArgonR primarily aims at building static webpages, without the need of shiny or server part. However, it can be also used within shiny packages such as argonDash, a bootstrap4 shiny dashboard |
| 9 | Create web dashboards | [argonDash](https://github.com/RinteRface/argonDash) | RinteRface | Same as ArgonR, However, it can be also used within shiny packages, a bootstrap4 shiny dashboard |
| 10 | UI -  Beautiful Shiny | [shiny.semantic](https://appsilon.com/open-source-overview/) | Appsilon | In short, It’s a library that makes it easy to wrap Shiny with Semantic UI components. It attaches all semantic external files which are stored on our CDN for improved loading speeds. The package has many popular components wrapped by default, but it is not hard to extend it even more by creating specific custom components that you may need. For that purpose, it has a universal input binding method for your custom user interfaces which enables you to create various types of inputs. |
| 11 | UI -  Beautiful Shinydashboard | [semantic.dashboard](https://appsilon.com/open-source-overview/) | Appsilon | The syntax is compatible with the ‘shinydashboard’  If you haven’t used a dashboard as a base for your app before, give it a try. It allows you to easily structure your app, making it more user-friendly. You divide your UI between 3 sections – header, a sidebar for navigation, and a dashboard body for displaying main content. With our package, you can also easily test out different Semantic-UI themes with a line of code. You can find some of those on [semantic forest](https://semantic-ui-forest.com/themes/). If you need more flexibility in styling you can also customize it to your liking with CSS |
| 12 | Introduce URL routing to Shiny app | [shiny.router](https://appsilon.com/open-source-overview/) | Appsilon | Shiny.router is a tool that will help you introduce URL routing to your Shiny app. |
| 13 | Create multi-language apps | [shiny.i18n](https://appsilon.com/open-source-overview/) | Appsilon | i18n supports translations in JSON or CSV format. You can have all your languages and translations in one file, but if it is needed, languages can be separated between different files for each language. This allows you to spread the translation process which may be crucial if you are going to translate into multiple languages. |
| 14 | UI –  Display dev info | [shiny.info](https://appsilon.com/open-source-overview/) | Appsilon | shiny.info introduces boxes with simple diagnostic information for developers that show up in the corner of your Shiny app. You can display things like:   * Loader that shows up during long computation – busy() * App version from global variable VERSION – version() * Box with branding – powered\_by(“Company”, “#Link”) * Git information – git\_info() * Custom message or anything else to help with your development – display(“Some diagnostic information”) |
| 16 | UI – Swipe elements | [shinyswipr](https://github.com/nstrayer/shinyswipr) | Nick Strayer |  |
| 17 | Create dashboards | [flexdashboard](https://rmarkdown.rstudio.com/flexdashboard/) | RStudio | Easy interactive dashboards for R |
| 18 | Create dashboards & reports in app | [Rmarkdown](https://rmarkdown.rstudio.com/) | RStudio | Easy static or interactive html, pdf, word, etc reports to communicate effectively your outputs |
| 19 |  | [shinysense](https://github.com/nstrayer/shinysense) (Not on CRAN yet, not totally stable) | Nick Strayer | Currently the package supports the following 'senses'.  ***Touch :***   * ***drawr*** & ***shinydrawr***: Draws a line chart that obscures the end of the results, the user then draws what they think the rest of the chart is and then the rest of the chart is revealed.   + Blatantly stolen from the New York Times article [You Draw It: What Got Better or Worse During Obama’s Presidency](https://www.nytimes.com/interactive/2017/01/15/us/politics/you-draw-obama-legacy.html). * shinyswipr: Embeds a card that can be swiped in different directions, the swipe direction is returned to shiny. |
| *Vision:*  * ***shinyviewr:*** Record images from a webcam or any other camera connected to browser viewing app. |
| *Hearing:*  * ***shinylistenr:*** Records audio on a button press and returns the fast-fourier transformed signal to the server. |
| *Motion:*  * ***shinymovr: Capture and return accelerometer data from your phone or tablet.*** * ***Used in*** [***this cast spells shiny app***](https://jhubiostatistics.shinyapps.io/cast_spells/) ***to classify spell cast motions using deep learning*** |
| 20 | Prod scale | [golem](https://github.com/ThinkR-open/golem) | ThinkR | Golem is an opinionated framework for building production-grade shiny applications.  Allow creating Shiny apps as R packages, also to Dockerize them and deploy easily to shinyproxy.io |
| 21 | Prod – scaling –Testing | [shinyTest](https://rstudio.github.io/shinytest/) | RStudio | After you get your Shiny application to a state where it works, it’s often useful to have an automated system that checks that it continues to work as expected. There are many possible reasons for an application to stop working. These reasons include:   * An upgraded R package has different behavior. (This could include Shiny itself!) * You make modifications to your application. * An external data source stops working, or returns data in a changed format.   One way to detect these problems is with manual testing – in other words, by having a person interact with the app in a browser – but this can be time-intensive, inconsistent, and imprecise. Having automated tests can alert you to these kinds of problems quickly and with almost zero effort, after the tests have been created |
| 22 | Prod – scaling –Testing | [shinyloadtest](https://rstudio.github.io/shinyloadtest/) | RStudio | **shinyloadtest** is an R package used to generate recordings and analyze results. You should install it on your development machine.  The **shinyloadtest** package and the accompanying [***shinycannon***](https://github.com/rstudio/shinycannon) software enable load testing deployed Shiny applications. |
| 23 | Performance Testing | [Profvis](https://rstudio.github.io/profvis/) | RStudio |  |
| 24 | Performance | Plot Cashing | RStudio | A functions series in Shiny, not a package. **Dramatically speed up repeated plots** |
| 25 | Performance | [Async](https://cran.r-project.org/web/packages/promises/vignettes/overview.html) (promises package) | RStudio | A method not a package. |
| 26 | UI – Prototyping | [shinipsum](https://github.com/ThinkR-open/shinipsum) | ThinkR | The goal of {shinipsum} is to provide random shiny elements for easiest shiny app prototyping, so that you can focus on building the frontend before building the backend. |
| 27 | Save time in developing apps | [shinysnippets](https://github.com/ThinkR-open/shinysnippets) | ThinkR | The goal of shinysnippets is to save development time while taking advantage of Rstudio snippets for Shiny applications. |
| 28 | UI – Prototyping | [fakir](https://thinkr-open.github.io/fakir/) | ThinkR | Generate data easily to train functions when prototyîng a shiny app |
| 29 | Mobile | [shinymobile](https://rinterface.github.io/shinyMobile/) | RinteRface | Develop outstanding {shiny} apps for iOS, Android, desktop as well as beautiful {shiny} gadgets. {shinyMobile} is built on top of the latest [***Framework7***](https://framework7.io) template. |