

## **Software Details (for developers)**

**Updated May 12<sup>th</sup>, 2022.**

### **Description:**

The purpose of this RShiny application is to be able to browse reference libraries and upload query spectra to search against these libraries and generate reports in a printer-friendly format. The application has four main tabs: the Data Search, Libraries, Scoring and About. The Data Search tab is where users can upload query spectra obtained from DART-MS to be searched against a reference library available through the application. The Libraries tab allows users to browse through Reference libraries and see the spectra associated with different compounds. Scoring walks through the details of how scores are computed in the search. Legal disclaimers and contact information can be found on the About tab. In the future a Library builder tab may be offered.

### **System Requirements and Prerequisites:**

Windows (for standalone use) or any system running R version 4.1 or newer.

### **Installing:**

This software is a standalone application that makes use of Chrome and RStudio. All executables and source code are included for offline use. For viewing of the actual source code, a recent version of RStudio may be used to navigate the backend of the software.

## Usage:

This software runs as a standalone independent of RStudio. Nonetheless, the application can be developed within RStudio. The following commands may be useful.

# Load an external package

```
library(shiny)
```

# Set the working directory to the path above the shiny/ folder in order to use runApp.R

```
setwd("Directory path above the shiny/ folder")
```

# Load a function you have written into the workspace

```
source("path to your file/filename.R", local = TRUE)
```

# Create a reactive variable for use within the shiny app and add fields

```
variablename <- reactiveValues()
```

```
variablename$field1 <- 1
```

# Create an observable event and the function to handle that event using an input tag

```
observeEvent(input$inputName, {Actions defined here})
```

# Write data to a .csv file for later usage and saving the current workspace

```
write.csv(data.frame(variables to be saved here), filename))
```

# Access a reactive value without taking a dependency on it (useful for updating non-reactive values during a reactive context.

```
Isolate({Code to be executed without dependency})
```

# Specific components of the code can be searched within the RStudio environment as such and entering specific search phrase:

```
Ctrl + Shift+ F
```

## **Glossary of Important Source Code and Data Files (details linked)**

### **NIST-NIJ-DIT/**

runApp\_DIT.R

shinyLastSession.csv

### **shiny\_DIT/**

app.R

### **Libraries/**

### **source/**

asm\_appUI.R

asm\_externalPackages.R

asm\_Header.R

asm\_sst\_appServer.R

### **Functions/**

asm-bpNormalizer.R

asm-clc.R

asm-hiRes2lowRes.R

asm-mass-diff.R

asm-PeaksExplained.R

asm-revMatchFactor.R

asm-spec2dt.R

asm-spec2dt\_ref.R

asm-specImport.R

asm-targetMolecules.R

asm\_jsp2txt.R

### **GUI/**

asm\_GUI\_DBV.R

asm\_GUI\_LB.R

asm\_sst\_GUI\_DST.R

sst\_GUI\_ABOUT.R

## **ServerFunctions/**

getAxisLimits.R

init\_rv.R

## **DataSearch/**

Adv\_Epsilon\_0.R

Adv\_LowRes.R

Adv\_SearchType.R

Adv\_TargetMinAb.R

expanded\_Fields.R

expanded\_PeakList.R

expanded\_Plot.R

expanded\_Plot\_Annotated.R

expanded\_QueryPlots.R

expanded\_Targets.R

interactive\_PeakList.R

interactive\_QueryPlots.R

interactive\_Targets.R

mainPanel\_downloadpresets.R

mainPanel\_LoadParams.R

mainPanel\_RestoreDefaults.R

mainPanel\_Toggle\_DST.R

Other\_HelperFunctions.R

Other\_SessionEnd.R

sidebar\_ClearSearch.R

sidebar\_FileInputs.R

sidebar\_RefLibSelect.R

sidebar\_Trigger\_Button.R

sidebar\_Trigger\_Button2.R

## **DataViewer/**

expanded\_LibraryPlots.R  
expanded\_PeakList\_DBV.R  
interactive\_LibraryPlots.R  
interactive\_LibraryPlotsTab.R  
interactive\_PeakList\_DBV.R  
interactive\_PeakListTab\_DBV.R  
mainPanel\_Library.R  
mainPanel\_structurePlotter.R  
mainPanel\_toggle\_DBV.R  
sidebar\_Library.R  
sidebar\_LibraryRows.R  
sidebar\_RefLibSelect.R

## External Packages

1. Shiny: rShiny package used for making interactive web applications
2. Shinjs: shinyjs used for integrating javascript in web app
3. Shinythemes: shinythemes used for customizing themes and text of the web app
4. Data.table: data.table is used for creating data frames for data organization
5. Devtools: used for github instsall
6. DT: sed for creating datatables within rshiny framework. Allows for more customizability than dataframes
7. Httr: used for working with URLs and HTTP within rshiny framework
8. Plotly: plotly is used for generating plots with annotations and customizations
9. Ggplot2: allows for interactive plots with custom themes
10. Ggrepel: used for plot annotations that are self repelling
11. Ggtext: element\_markdown theme for plot text customization
12. Stringr: reading and manipulation of string data

## Documentation

1. runApp\_DIT.R - The purpose of this code is to clear the working environment and run the shiny application after installing any external packages
2. shinyLastSession.csv – contains presets from the previous session to load on the next session
3. app.R – Creates the actual shiny App after loading asm\_Header.R, asm\_appUI.R, and asm\_sst\_appServer.R
4. asm\_appUI.R - Load the GUI tabs for Database viewer and search as well as the about tabs. These components are packaged into the shinyUI appUI as part of the navbarPage
5. asm\_externalPackages.R - Install packages that the source code depends on for running. This code is run at the initial code startup and if any packages are missing from the environment they are installed here
6. asm\_Header.R - Initialize helper functions, library data, previous session data, disclaimer messages and dev mode. This program is also where dev\_mode and lb\_mode are turned on and off
7. asm\_sst\_appServer.R - The purpose of this program is to define the server end of the shiny app, the other component being the app UI. This program uses helper functions, inputs, outputs, observeEvents, eventReactivities which are organized accordingly
8. asm-bpNormalizer.R
9. asm-clc.R
10. asm-hiRes2lowRes.R
11. asm-mass-diff.R
12. asm-PeaksExplained.R
13. asm-revMatchFactor.R
14. asm-spec2dt.R
15. asm-spec2dt\_ref.R
16. asm-specImport.R
17. asm-targetMolecules.R
18. asm\_jsp2txt.R
19. asm\_GUI\_DBV.R - The purpose of this program is to define the UI end of the shiny server app. This program specifically defines the Libraries tab where users can search through different reference libraries and retrieve spectra and structure information about different compounds.
20. asm\_GUI\_LB.R - This program is future work for library building

21. `asm_sst_GUI_DST.R` - This program defines the UI for the Data search tab of the DART-MS Search software. Users can use this tab to upload query spectra and search the reference libraries, as well as toggle between printer friendly and interactive views of the data.
22. `sst_GUI_ABOUT.R` - This program defines the About tab of the DART-MS Search application. Users can use this tab to find disclaimer and contact information.
23. `getAxisLimits.R` – Helper function that determines the maximum range of the three spectra passed in for x and y axis limits.
24. `init_rv.R` – initializes the `reactiveValues()` object 'rv' and all subfields, as well as other initializations necessary for loading previous session data and expanded view fields.
25. `Adv_Epsilon_0.R` – Advanced setting parameter to update rv epsilon value
26. `Adv_LowRes.R` - Advanced setting parameter to respond to input for lowres
27. `Adv_SearchType.R` - Disable `target_min_ab` if Pure compound selected
28. `Adv_TargetMinAb.R` - Set the `target_min_ab` for the shiny app
29. `expanded_Fields.R` - This program creates a table in Expanded view of DST for current parameters
30. `expanded_PeakList.R` - The purpose of this program is to define the expanded view peak lists. This is done by output `$expandedPeakList1`, 2 and 3 as well as a helper function which returns the data based on the currently selected tab from the interactive view
31. `expanded_Plot.R`- This function creates expanded view plots but without annotations For 60 and 90V
32. `expanded_Plot_Annotated.R` - This function is used to create annotated plots in expanded view using outputs at bottom of this script for 3 spectra plots
33. `expanded_QueryPlots.R` - This program creates the expanded view for query plots 30, 60, 90 V and any combination of these spectra for dynamic number of file uploads It makes use of `ExpandedPlot_Annotated.R` for the 30V spectrum and `ExpandedPlot.R` for the 60 and 90V spectra
34. `expanded_Targets.R` - Show the targets in interactive view using `renderUI`
35. `interactive_PeakList.R` - This program contains two outputs, `peakListUI` and `interactivePeakList`. These outputs create the three tabs to be selected from the peak list and the peak list itself for the interactive view of the search tab
36. `interactive_QueryPlots.R` - This program contains outputs for `QueryPlotsUI` and the `QueryPlots` for display of the query spectra tabs and their plots for interactive view in the search tab
37. `interactive_Targets.R` - Show the targets in interactive view using `renderUI`
38. `mainPanel_downloadpresets.R` - This function saves a .csv of the currently selected parameters
39. `mainPanel_LoadParams.R` - This function loads saved parameters from user uploaded parameter profile using helper function `file.choose2` for file selection
40. `mainPanel_RestoreDefaults.R` - This function restores all settings to default without requiring closing the program. The `shinyLastSession.csv` is also deleted and previous search cannot be restored



41. `mainPanel_Toggle_DST.R` - Show/hide certain features of DST tab when in Interactive vs expanded
42. `Other_HelperFunctions.R` – Helper functions associated with target analysis and error messages stored here
43. `Other_SessionEnd.R` - What to do once app is closed is handled here, including downloading the current sessions parameters to be pre-populated on the next session via 'LastSession.csv'
44. `sidebar_ClearSearch.R` - This function clears the current search, resets file inputs and initializes associated variables with search back to start
45. `sidebar_FileInputs.R` - This function handles the three file input datapaths for the sidebar of the search tab
46. `sidebar_RefLibSelect.R` - RefLibSelect dropdown menu for DST tab. Must use `input$RefLibSelect_DST` to avoid conflict with RefLibSelect on DBV tab. Use `rv_RefLibSelected` to isolate from reactive values
47. `sidebar_Trigger_Button.R` - `eventReactive` responds to `input$DartSearch` to calculate plots and targets When making changes to `trigger_button` please copy all lines after line 11 to ensure functionality not damaged in Advanced search tab
48. `sidebar_Trigger_Button2.R` - Trigger button2 is identical to `trigger_button` except that it uses `input$DartSearchAdv` inputs for the Advanced panel in DST. Use `currtrigger = 2` to refer to this `trigger_button` versus the Query Spectra panel
49. `expanded_LibraryPlots.R` - This program creates the 3 plots in expanded view for the currently selected compound in the Libraries tab. A helper function is included at the bottom of this program to eliminate some redundancy. There are three outputs, one for each of the plots in expanded view
50. `expanded_PeakList_DBV.R` - This program creates the expanded view peak lists for the 30, 60, and 90V spectrums in the Libraries tab. There are three outputs included, one for each of the peak lists. There is also a helper function at the bottom which returns the peak list data (`bpn_data`) for each of the spectra depending on which tab was last selected in the interactive view
51. `interactive_LibraryPlots.R` - The purpose of this program is to create the library plots and the UI tabs (30, 60, and 90V tabs) for the interactive view
52. `interactive_LibraryPlotsTab.R` - Update which tab is selected for plots in Libraries tab interactive view
53. `interactive_PeakList_DBV.R` - This program has two outputs. The first output creates the UI for the peak list in interactive view. The second output creates the peak list itself for each tab of the peaklist UI
54. `interactive_PeakListTab_DBV.R` - Update which tab is selected for peak list in Libraries tab interactive view
55. `mainPanel_Library.R` - The purpose of this function is to create the main panel table for library data to be displayed for the currently selected compound. This code also makes use of a Helper function at the bottom that sanitizes the text function to enable for breaks, subscripts, displaying of greek symbol, and handling of href

56. `mainPanel_structurePlotter.R` - This program creates the chemical structure to be displayed for each compound
57. `mainPanel_toggle_DBV.R` - This program creates a toggle which Shows/hides certain elements of DBV when in interactive vs expanded view
58. `sidebar_Library.R` - The purpose of this code is to load the library data for the Libraries sidebar. This is done by loading each column of the data displayed and rendering it into a `DT::DataTable` with the help of `makesubscript`
59. `sidebar_LibraryRows.R` - This `observeEvent` is used to update the number of rows displayed in the sidebar based on the dropdown menu selection for number of rows to display
60. `sidebar_RefLibSelect.R` - `RefLibSelect` dropdown menu for Libraries allows user to select which library to use viewer in. Warning: Must use `input$RefLibSelect` to avoid conflict with `RefLibSelect` on DST tab

**Support:**

For more help and information please contact [dartdata@nist.gov](mailto:dartdata@nist.gov)

**Authors and Acknowledgement:**

This software was developed by employees of the National Institute of Standards and Technology (NIST), an agency of the Federal Government and is being made available as a public service. Pursuant to title 17 United States Code Section 105, works of NIST employees are not subject to copyright protection in the United States. This software may be subject to foreign copyright. Permission in the United States and in foreign countries, to the extent that NIST may hold copyright, to use, copy, modify, create derivative works, and distribute this software and its documentation without fee is hereby granted on a non-exclusive basis, provided that this notice and disclaimer of warranty appears in all copies.

**License:**

This Web site is funded, in part, through a grant from the Office of Forensic and Investigative Sciences, Office of Justice Programs, U.S. Department of Justice. Neither the U.S. Department of Justice nor any of its components operate, control, are responsible for, or necessarily endorse, this Web site (including, without limitation, its content, technical infrastructure, and policies, and any services or tools provided).

THE SOFTWARE IS PROVIDED 'AS IS' WITHOUT ANY WARRANTY OF ANY KIND, EITHER EXPRESSED, IMPLIED, OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, ANY WARRANTY THAT THE SOFTWARE WILL CONFORM TO SPECIFICATIONS, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND FREEDOM FROM INFRINGEMENT, AND ANY WARRANTY THAT THE DOCUMENTATION WILL CONFORM TO THE SOFTWARE, OR ANY WARRANTY THAT THE SOFTWARE WILL BE ERROR FREE. IN NO EVENT SHALL NIST BE LIABLE FOR ANY DAMAGES, INCLUDING, BUT NOT LIMITED TO, DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES, ARISING OUT OF, RESULTING FROM, OR IN ANY WAY CONNECTED WITH THIS SOFTWARE, WHETHER OR NOT BASED UPON WARRANTY, CONTRACT, TORT, OR OTHERWISE, WHETHER OR NOT INJURY WAS SUSTAINED BY PERSONS OR PROPERTY OR OTHERWISE, AND WHETHER OR NOT LOSS WAS SUSTAINED FROM, OR AROSE OUT OF THE RESULTS OF, OR USE OF, THE SOFTWARE OR SERVICES PROVIDED HEREUNDER.