

Package ‘shinylive’

March 31, 2026

Title Run 'shiny' Applications in the Browser

Version 0.4.0

Description Exporting 'shiny' applications with 'shinylive' allows you to run them entirely in a web browser, without the need for a separate R server. The traditional way of deploying 'shiny' applications involves in a separate server and client: the server runs R and 'shiny', and clients connect via the web browser. When an application is deployed with 'shinylive', R and 'shiny' run in the web browser (via 'webR'): the browser is effectively both the client and server for the application. This allows for your 'shiny' application exported by 'shinylive' to be hosted by a static web server.

License MIT + file LICENSE

URL <https://posit-dev.github.io/r-shinylive/>,
<https://github.com/posit-dev/r-shinylive>

BugReports <https://github.com/posit-dev/r-shinylive/issues>

Imports archive, brio, cli, fs, gh, glue, httr2 (>= 1.0.0), jsonlite, lifecycle, pkgdepends, rappdirs, renv, rlang (>= 1.1.0), tools, whisker, withr

Suggests httpuv (>= 1.6.12), pkgcache, spelling, testthat (>= 3.0.0)

Config/Needs/website tidyverse/tidytemplate

Config/testthat/edition 3

Config/usethis/last-upkeep 2026-03-30

Encoding UTF-8

Language en-US

RoxygenNote 7.3.3

NeedsCompilation no

Author Barret Schloerke [aut, cre] (ORCID: <https://orcid.org/0000-0001-9986-114X>),
Winston Chang [aut] (ORCID: <https://orcid.org/0000-0002-1576-2126>),
George Stagg [aut],

Garrick Aden-Buie [aut] (ORCID:
<https://orcid.org/0000-0002-7111-0077>),
 Posit Software, PBC [cph, fnd] (ROR: <https://ror.org/03wc8by49>)

Maintainer Barret Schloerke <barret@posit.co>

Repository CRAN

Date/Publication 2026-03-31 19:20:02 UTC

Contents

assets_download	2
assets_install_copy	3
export	4
quarto_ext	6

Index **10**

assets_download	<i>Manage shinylive assets</i>
-----------------	--------------------------------

Description

Helper methods for managing shinylive assets.

Usage

```
assets_download(
  version = assets_version(),
  ...,
  dir = assets_cache_dir(),
  url = assets_bundle_url(version)
)

assets_ensure(
  version = assets_version(),
  ...,
  dir = assets_cache_dir(),
  url = assets_bundle_url(version)
)

assets_cleanup(..., dir = assets_cache_dir())

assets_remove(versions, ..., dir = assets_cache_dir())

assets_info(quiet = FALSE)

assets_version()
```

Arguments

version	The version of the assets to download.
...	Ignored.
dir	The asset cache directory. Unless testing, the default behavior should be used.
url	The URL to download the assets from. Unless testing, the default behavior should be used.
versions	The assets versions to remove.
quiet	In assets_info(), if quiet = TRUE, the function will not print the assets information to the console.

Value

assets_version() returns the version of the currently supported Shinylive.

All other methods return invisible().

Functions

- assets_download(): Downloads the shinylive assets bundle from GitHub and extracts it to the specified directory. The bundle will always be downloaded from GitHub, even if it already exists in the cache directory (dir=).
- assets_ensure(): Ensures a local copy of shinylive is installed. If a local copy of shinylive is not installed, it will be downloaded and installed. If a local copy of shinylive is installed, its path will be returned.
- assets_cleanup(): Removes local copies of shinylive web assets, except for the one used by the current version of **shinylive**.
- assets_remove(): Removes a local copies of shinylive web assets.
- assets_info(): Prints information about the local shinylive assets that have been installed. Invisibly returns a table of installed asset versions and their associated paths.
- assets_version(): Returns the version of the currently supported Shinylive assets version. If the SHINYLIVE_ASSETS_VERSION environment variable is set, that value will be used.

assets_install_copy *Install shinylive assets from from a local directory*

Description

Helper methods for testing updates to shinylive assets.

Usage

```
assets_install_copy(  
  assets_repo_dir,  
  ...,  
  dir = assets_cache_dir(),  
  version = package_json_version(assets_repo_dir)  
)  
  
assets_install_link(  
  assets_repo_dir,  
  ...,  
  dir = assets_cache_dir(),  
  version = package_json_version(assets_repo_dir)  
)
```

Arguments

assets_repo_dir	The local repository directory for shinylive assets (e.g. <code>posit-dev/shinylive</code>)
...	Ignored.
dir	The asset cache directory. Unless testing, the default behavior should be used.
version	The version of the assets being installed.

Value

All method return `invisible()`.

Functions

- `assets_install_copy()`: Copies all shinylive assets from a local shinylive repository (e.g. `posit-dev/shinylive`). This must be repeated for any change in the assets.
- `assets_install_link()`: Creates a symlink of the local shinylive assets to the cached assets directory. After the first installation, the assets will be the same as the source due to the symlink.

See Also

[assets_download\(\)](#), [assets_ensure\(\)](#), [assets_cleanup\(\)](#)

export

Export a Shiny app to a directory

Description

This function exports a Shiny app to a directory, which can then be served using `httpuv`.

Usage

```

export(
  appdir,
  destdir,
  ...,
  subdir = "",
  quiet = getOption("shinylive.quiet", !is_interactive()),
  wasm_packages = NULL,
  package_cache = TRUE,
  max_filesize = NULL,
  assets_version = NULL,
  template_dir = NULL,
  template_params = list(),
  verbose = deprecated()
)

```

Arguments

appdir	Directory containing the application.
destdir	Destination directory.
...	Ignored
subdir	Subdirectory of destdir to write the app to.
quiet	Suppress console output during export. Follows the global shinylive.quiet option or defaults to FALSE in interactive sessions if not set.
wasm_packages	Download and include binary WebAssembly packages as part of the output app's static assets. Logical, defaults to TRUE. The default value can be changed by setting the environment variable SHINYLIVE_WASM_PACKAGES to TRUE or 1 to enable, FALSE or 0 to disable.
package_cache	Cache downloaded binary WebAssembly packages. Defaults to TRUE.
max_filesize	Maximum file size for bundling of WebAssembly package assets. Parsed by <code>fs::fs_bytes()</code> . Defaults to "100M". The default value can be changed by setting the environment variable SHINYLIVE_DEFAULT_MAX_FILESIZE. Set to Inf, NA or -1 to disable.
assets_version	The version of the Shinylive assets to use in the exported app. Defaults to <code>assets_version()</code> . Note, not all custom assets versions may work with this release of shinylive . Please visit the shinylive asset releases website to learn more information about the available assets_version values.
template_dir	Path to a custom template directory to use when exporting the shinylive app. The template can be copied from the shinylive assets using: <code>fs::path(shinylive::assets_dir(), "export_template")</code> .
template_params	A list of parameters to pass to the template. The supported parameters depends on the template being used. Custom templates may support additional parameters (see <code>template_dir</code> for instructions on creating a custom template or to find the current shinylive assets' templates).

With shinylive assets > 0.4.1, the default export template supports the following parameters:

1. `title`: The title of the app. Defaults to "Shiny app".
2. `include_in_head`, `include_before_body`, `include_after_body`: Raw HTML to be included in the `<head>`, just after the opening `<body>`, or just before the closing `</body>` tag, respectively.

`verbose` **[Deprecated]** Use `quiet` instead.

Value

Nothing. The app is exported to `destdir`. Instructions for serving the directory are printed to `stdout`.

Examples

```
app_dir <- system.file("examples", "01_hello", package = "shiny")
out_dir <- tempfile("shinylive-export")

# Export the app to a directory
export(app_dir, out_dir)

# Serve the exported directory
if (require(httpuv)) {
  httpuv::runStaticServer(out_dir)
}
```

quarto_ext

Quarto extension for shinylive

Description

Integration with <https://github.com/quarto-ext/shinylive>

Usage

```
quarto_ext(
  args = commandArgs(trailingOnly = TRUE),
  ...,
  pretty = is_interactive(),
  con = "stdin"
)
```

Arguments

args	Command line arguments passed by the extension. See details for more information.
...	Ignored.
pretty	Whether to pretty print the JSON output.
con	File from which to take input. Default: "stdin".

Value

Nothing. Values are printed to stdout.

Command arguments

The first argument must be "extension". This is done to match py-shinylive so that it can nest other sub-commands under the extension argument to minimize the api clutter the user can see.

CLI Interface:

- extension info
 - Prints information about the extension including:
 - * version: The version of the R package
 - * assets_version: The version of the web assets
 - * scripts: A list of paths scripts that are used by the extension, mainly codeblock-to-json
 - Example


```
{
  "version": "0.1.0",
  "assets_version": "0.2.0",
  "scripts": {
    "codeblock-to-json": "/<ASSETS_CACHE_DIR>/shinylive-0.2.0/scripts/codeblock-to-json.js"
  }
}
```
- extension base-htmldeps
 - Prints the language agnostic quarto html dependencies as a JSON array.
 - * The first html dependency is the shinylive service workers.
 - * The second html dependency is the shinylive base dependencies. This dependency will contain the core shinylive asset scripts (JS files automatically sourced), stylesheets (CSS files that are automatically included), and resources (additional files that the JS and CSS files can source).
 - Example


```
[
  {
    "name": "shinylive-serviceworker",
    "version": "0.2.0",
    "meta": { "shinylive:serviceworker_dir": "." },
    "serviceworkers": [
      {
```

```

      "source": "/<ASSETS_CACHE_DIR>/shinylive-0.2.0/shinylive-sw.js",
      "destination": "/shinylive-sw.js"
    }
  ]
},
{
  "name": "shinylive",
  "version": "0.2.0",
  "scripts": [{
    "name": "shinylive/load-shinylive-sw.js",
    "path": "/<ASSETS_CACHE_DIR>/shinylive-0.2.0/shinylive/load-shinylive-sw.js",
    "attrs": { "type": "module" }
  }],
  "stylesheets": [{
    "name": "shinylive/shinylive.css",
    "path": "/<ASSETS_CACHE_DIR>/shinylive-0.2.0/shinylive/shinylive.css"
  }],
  "resources": [
    {
      "name": "shinylive/shinylive.js",
      "path": "/<ASSETS_CACHE_DIR>/shinylive-0.2.0/shinylive/shinylive.js"
    },
    ... # [ truncated ]
  ]
}
]

```

- extension language-resources

- Prints the language-specific resource files as JSON that should be added to the quarto html dependency.

- * For r-shinylive, this includes the webr resource files

- * For py-shinylive, this includes the pyodide and pyright resource files.

- Example

```

[
  {
    "name": "shinylive/webr/esbuild.d.ts",
    "path": "/<ASSETS_CACHE_DIR>/shinylive-0.2.0/shinylive/webr/esbuild.d.ts"
  },
  {
    "name": "shinylive/webr/libRblas.so",
    "path": "/<ASSETS_CACHE_DIR>/shinylive-0.2.0/shinylive/webr/libRblas.so"
  },
  ... # [ truncated ]
]

```

- extension app-resources

- Prints app-specific resource files as JSON that should be added to the "shinylive" quarto html dependency.

- Currently, r-shinylive does not return any resource files.

– Example

```
[
  {
    "name": "shinylive/pyodide/anyio-3.7.0-py3-none-any.whl",
    "path": "<ASSETS_CACHE_DIR>/shinylive-0.2.0/shinylive/pyodide/anyio-3.7.0-py3-none-any.",
  },
  {
    "name": "shinylive/pyodide/appdirs-1.4.4-py2.py3-none-any.whl",
    "path": "<ASSETS_CACHE_DIR>/shinylive-0.2.0/shinylive/pyodide/appdirs-1.4.4-py2.py3-non",
  },
  ... # [ truncated ]
]
```

Index

`assets_cleanup (assets_download)`, 2
`assets_cleanup()`, 4
`assets_download`, 2
`assets_download()`, 4
`assets_ensure (assets_download)`, 2
`assets_ensure()`, 4
`assets_info (assets_download)`, 2
`assets_install_copy`, 3
`assets_install_link`
 (`assets_install_copy`), 3
`assets_remove (assets_download)`, 2
`assets_version (assets_download)`, 2
`assets_version()`, 5

`export`, 4

`fs::fs_bytes()`, 5

`quarto_ext`, 6