Tin: A Tcl Package Manager

Version 0.1

Alex Baker

https://github.com/ambaker1/tin

March 31, 2023

Installing and updating Tin

First install: download the latest release from GitHub, extract the files, and run the following code:

Example 1: Installing tin (first time)

Code:

```
source tin.tcl
tin extract tin
exit
```

Once installed, updating Tin is easy: just open up a Tcl interpreter and run the following code:

Example 2: Upgrading Tin

Code:

```
package require tin
tin install tin
```

Installing Tin-compatible packages

The command *tin extract* installs Tin-compatible packages from a local directory, and the command *tin install* installs Tin-compatible packages directly from GitHub. Both *tin extract* and *tin install* return the version number of the package installed.

```
tin extract $package <$src> <$requirement ...>
```

```
tin install $package <$requirement ...>
```

$package	Package name
$src	Directory to extract package from. Default current directory.
$requirement ...	Version requirements. See Tcl *package* documentation.

Tin comes pre-packaged with a list of packages available for installation. This list can be queried with *tin packages*. To add a tin-compatible list that is not on the list, use the command *tin add*.

```
tin packages
```

```
tin add $package $repo
```

$package	Package name
$repo	Github repository URL

Example 3: Adding a package to the Tin and installing it

Code:

```
package require tin
tin add foo https://github.com/username/foo
tin install foo
```

Requiring and importing packages

The Tin package also provides advanced tools for requiring and importing packages. The command *tin require* is similar to the Tcl command *package require*, but with the added feature that if the package is missing, it will try to install it with *tin install*. The command *tin import* additionally handles most use-cases of *namespace import*. Both *tin require* and *tin import* return the version number of the package imported.

```
tin require $package <$requirement ...>
```

$package	Package name
$requirement ...	Version requirements. See Tcl *package* documentation.

```
tin import <$patterns from> $package <$requirements> <as $namespace>
```

$patterns	List of commands or "glob" style patterns to import. Default "*", or all exported commands.
$package	Package name
$requirements	List of version requirements. See Tcl *package* documentation.
$namespace	Namespace to import into (default current namespace)

Example 4: Importing all commands from a package

Code:

```
package require tin
tin import foo
```

What makes a package Tin-compatible?

Tin-compatible packages must have a "tinstall.tcl" file which copies required files from the main repository folder to the Tcl library folder, represented by variables $src and $dir, respectively. Additionally, the "tinstall.tcl" file must contain a *tin provide* statement at the end of the file with the package name and version. If a Tin package requires other Tin packages, dependencies can be handled with the *tin depend* command.

tin provide $package $version

$package	Package name
$version	Version number (e.g. 3.1.4)

tin depend $package <$requirement ...>

$package	Package name
$requirement ...	Version requirements. See Tcl *package* documentation.

See the example below for the Tin package “bar 2.4” that requires the Tin package “foo 1.2”:

Example 5: Example “tinstall.tcl” file

Code:

```
tin depend foo 1.2
file copy [file join $src README.md] $dir
file copy [file join $src LICENSE] $dir
file copy [file join $src lib/bar.tcl] $dir
file copy [file join $src lib/pkgIndex.pdf] $dir
tin provide bar 2.4
```

Including a "tinstall.tcl" file will make the repository compatible with the *tin extract* command. To make it compatible with the *tin install* command, which allows for automatic installation from GitHub, the repository must also have release tags with the format “v0.0.0”.