

README

Penguin's eggs are generated and new birds are ready to fly...

[github](#) [sources](#)[www](#) [blog](#)[telegram](#) [group](#)[basket](#) [naked](#)[gdrive](#) [all](#)[sourceforge](#) [all](#)[npm](#) [v25.12.7](#)[Get it as](#) [ApplImage](#)

It took years of work to create the penguins-eggs, and I also incurred expenses for renting the site and subscribing to Google Gemini, for the artificial intelligence that is now indispensable.



Index

- [README](#)
- [Penguin's eggs are generated and new birds are ready to fly...](#)
- [Presentation](#)
- [Installation](#)
- [Introduction](#)
- [Technology](#)
- [Features](#)
- [Packages](#)
- [Usage](#)
- [Commands](#)
- [GUI](#)

- [Book](#)
- [Copyright and licenses](#)

Links

- [Blog](#)
- [Cook eggs in 5 minutes!](#)
- [Users guide](#)
- [Wardrobe users' guide](#)
- [FAQ](#)
- [Changelog](#)

Presentation

penguins-eggs is a system cloning and distribution remastering tool primarily designed for Linux distributions.

It's often used to create customized live ISO images or backups of a Linux system, allowing users to replicate or share their setup easily. Think of it as a way to "hatch" a new system from an existing one—hence the funny name.

Here's a breakdown of what it's used for:

- **Distribution Remastering:** If you're into crafting your own Linux distro (or a spin of an existing one), penguins-eggs simplifies the process. You can tweak an existing system, strip out or add components, and then package it as a new ISO for distribution.
- **Creating Custom Live ISOs:** It lets you generate a bootable live ISO from your system. You can share this ISO with others or use it to boot your setup on different hardware. It's like creating your own personalized Linux distro without starting from scratch.
- **System Backup and Cloning:** You can use penguins-eggs to create a snapshot of your current Linux system, including installed packages, configurations, and user data (if you choose). This can be handy for disaster recovery or migrating to a new machine.

Lightweight and Distro-Agnostic: It works across various Debian, Devuan, Ubuntu, Arch, Fedora, Almalinux, Rocky, OpenSUSE and Alpine derivatives so you're not locked into one ecosystem. Please consult the complete list of [SUPPORTED-DISTROS](#) to get more informations.

Installation

ApplImage

Get it as **ApplImage**

Although penguins-eggs has only recently been released as an ApplImage, this is perhaps the simplest method. All you need is to get the penguins-eggs ApplImage: open a browser at the url: <https://github.com/pieroproietti/penguins-eggs>, on the right side, find latest release, download the ApplImage penguins-eggs-25.x.x-1-x86_64.AppImage and launch it using `sudo`

The ApplImage will be automatically configured and renamed as `/usr/bin/eggs`. It's ready to use!

`eggs` love to get your system remastered as live ISO.

Before trying the ApplImage, depending on your distro, you need to install these packages:

- Alpine: `sudo apk add fuse`
- Arch/Manjaro: `sudo pacman -S fuse2`
- Debian/Devuan/Ubuntu: `sudo apt-get install fuse libfuse2`
- Fedora/RHEL: `sudo dnf install fuse fuse-libs`
- Opensuse: `sudo zypper install fuse fuse-libs`

ApplImage installation

penguins-eggs as an ApplImage, it can be installed on all supported distributions. Download it from github.com/pieroproietti or on appimage.github.io, then run the following commands:

```
$ chmod +x penguins-eggs-25.11.29-1-x86_64.AppImage
$ sudo ./penguins-eggs-25.11.29-1-x86_64.AppImage
```

Note that the ApplImage version and the native version consist of exactly the same code and install/remove the same native dependencies on every distro.

Native packages

Try the new repos for [Debian/Devuan/Ubuntu](#), [Fedora](#), [Enterprise linux](#) and [OpenSUSE](#).

penguins-eggs-25.7.x installation

penguins-eggs-25.7.x depends on `nodejs >18`, which is not directly available on all distros. We can rely on [nodesource](#) for adding them.

using fresh-eggs

It's the most practical way and is suitable for all [supported distros](#).

[fresh-eggs](#) configure automatically nodesource when need, and install the right package for your distro. Copy and paste, the follow commands:

```
git clone https://github.com/pieroproietti/fresh-eggs
cd fresh-eggs
sudo ./fresh-eggs.sh
```

And follow instructions.

Manual installation

AlmaLinux, RockyLinux

Since penguins-eggs-25.7.14 rpm packages for Enterprise Linux [el9](#) are availables.

AlpineLinux

I maintain a package for Alpine Linux. Its repository is [penguins-alpine](#) and related packages can be downloaded from [penguins-eggs.net](#) or the [sourceforge page](#).

Arch

Just download and install penguins-eggs-25.x.x using [fresh-eggs](#).

Debian, Devuan, Ubuntu

Just download and install penguins-eggs-25.x.x using [fresh-eggs](#).

Debian 10 Buster, Debian 11 Bullseye

Before to install penguins-eggs-25.x.x, add the repos from nodesource. Follow these [instructions](#) to get nodejs>18 available.

Debian 9 Stretch, Ubuntu 18.04 Bionic

Use the package penguins-eggs-25.x.x-bionic-x, which is compiled against node16. Follow these [instructions](#) to get nodejs>16 available.

Fedora

Since penguins-eggs-25.7.14 rpm packages for [fedora42](#) are availables.

Manjaro

penguins-eggs is on Manjaro community repo, so just: `sudo pacman install penguins-eggs`.

Openmanba

Openmamba is an italian distribution based on rpm and using dnf as package manager, it is really up-to-date, penguins-eggs is available as rpm package.

OpenSUSE

Since penguins-eggs-25.7.14 rpm packages for [OpenSUSE](#) are availables.

Introduction

penguins-eggs is the package name of eggs a console tool, under continuous development, that allows you to remaster your system and redistribute it as live images on USB sticks or via PXE.

The concept behind Penguins' Eggs stems from the idea of "reproduction" and "population selection" applied to operating systems. During the era of popular remastering programs like Remastersys and Systemback, both of which experienced maintenance issues and were eventually abandoned, the need for a new, modern tool became evident.

The inspiration for Penguins' Eggs led to the development of a new tool written in a modern, cross-distribution language, utilizing its own packaging system.

Initially built with javascript and then switched to Typescript as the main development language, the design of the tool resembles an egg production process, consisting of operations such as produce to create the eggs and hatch to install them (I changed a bit later: from hatch to install, then to krill as a tribute to calamares tribes).

Other commands follow: like kill for removing produced ISOs, calamares for configuring the graphical installer, mom as interactive help, dad to configure eggs, status, tools, etc.

Considered a work-in-progress, the ultimate goal of Penguins Eggs is to allow the creation of live ISOs from an installed system and their use for system installations. It can be used both to create images on USB devices - even [Ventoy](#) - and as a PXE server to distribute the image itself over a local network. Inspired by the behavior of the cuckoo bird, which depends on others to hatch its eggs.

Written primarily in TypeScript, Penguins' Eggs is designed to be compatible with various Linux distributions, despite differences in package managers, file paths, and more.

The tool currently supports Debian, Devuan, Ubuntu, Arch, Manjaro, and their derivatives, across multiple architectures including amd64, i386, and arm64.

From the release of version 9.6.x, Penguins' Eggs is available as a [Debian package](#), then was extended to Arch Linux creating the relative [PKGBUILD](#). Nowadays, eggs can remaster AlmaLinux, Fedora, openmamba, openSUSE, and RockyLinux too.

Penguins Eggs caters to a wide range of systems including PCs, older machines, and single-board ARM systems like the Raspberry Pi, across amd64, i386, and arm64 architectures.

For more information and updates, visit the [Penguins Eggs' official website](#).

Tip

eggs is an actively developed console tool designed to help you customize and distribute your system as live images on USB sticks or through PXE. By using this tool, you can remaster your system according to your preferences.

Tip

By default, eggs completely removes the system's data and users. However, it also offers the option to remaster the system while including the data and accounts of existing users. This can be done using the `--clone` flag. Additionally, you can preserve the users and files by storing them in an encrypted LUKS file within the resulting ISO file, which can be achieved with the `--cryptedclone` flag.

Tip

The resulting live system can be easily installed using either the calamares installer or the internal TUI krill installer. Furthermore, using krill, you can easily get an unattended installation, utilizing `--unattended` and various [krill](#) flags.

Tip

One interesting feature of eggs is its integration with the penguins-wardrobe. This allows you to create or utilize scripts to switch between different configurations. For example, you can start with a bare version of the system, featuring only a command-line interface (CLI), and then easily transition to a full graphical user interface (GUI) or server configurations.

Note

For more information and customization options, you can explore penguins-wardrobe, a related project. You can fork it and adapt it to meet your specific

needs.

See [penguins-wardrobe](#), fork it, and adapt it to your needs.

Technology

eggs is primarily written in TypeScript and is designed to be compatible with various Linux distributions. While there may be differences in package managers, paths, and other aspects, the underlying programs used to build the live system are generally the same.

Currently, eggs supports several Linux distributions, including [AlmaLinux](#), [Arch](#), [Debian](#), [Devuan](#), [LinuxMint](#), [Manjaro](#), [openmamba](#), [openSuSE](#), [Ubuntu](#), [RockyLinux](#) and [derivatives](#).

It also caters to different architectures, namely amd64, i386, and arm64.

Starting from version 9.6.x, penguins-eggs is released as a Debian package, available for amd64, i386, and arm64 architectures. This allows it to support a wide range of PCs, including older machines, as well as single-board ARM systems like the Raspberry Pi. You can learn more about this release in the article titled Triple Somersault! [Triple somersault!](#).

For more information on the supported distributions and architectures, you can visit the blog [blog](#).

Additionally, you can find examples of remastered ISO images created with eggs on the project's SourceForge page [sourceforge page of the project](#).

Features

Penguins-eggs is a versatile tool that offers an array of features and benefits for Linux users. Whether you want to create an installable ISO from your current Linux system or explore various customization options,

Penguins-eggs has got you covered. To get started with Penguins-eggs, you'll need to install it on your Linux distribution. The tool supports a wide range of Linux distributions and their major derivatives, including Arch, Debian, Devuan, Manjaro, Ubuntu, and more. Additionally, you can easily add support for additional derivatives, expanding the tool's capabilities even further.

1. **Fast and Efficient** Penguins-eggs is designed to be fast and efficient. Unlike traditional methods that involve copying the entire file system, Penguins-eggs utilizes livefs, which allows for instant acquisition of the live system. By default, the tool.

2. **Supports Compression Algorithm** Employs the zstd compression algorithm, significantly reducing the time required for the process, often up to 10 times faster. When creating an installable ISO.
3. **Supports Clone** Penguins-eggs provides various options to suit your needs. With the --clone flag, you can preserve the data and accounts of unencrypted users, ensuring a seamless experience for users accessing the live system. Moreover, you can opt for a crypted clone, where user data and accounts are saved in an encrypted LUKS volume within the ISO image, enhancing security and privacy.
4. **Cuckoo and PXE boot** In addition to ISO creation, Penguins-eggs offers a unique feature called Cuckoo. By starting Cuckoo from the live system, you can set up a PXE boot server, making it accessible to all computers on the network. This functionality opens up possibilities for network booting and streamlined deployment. Penguins Eggs Linux ushers in a new era of innovation and convenience with its groundbreaking default feature, Cuckoo live network boot, which transforms any computer running Penguins Eggs into a PXE (Preboot eXecution Environment) boot server. This revolutionary paradigm of network booting and seamless deployment underscores Penguins Eggs Linux's commitment to redefining the parameters of accessibility and efficiency within the realm of Linux distributions.
5. **Supports Both TUI/GUI Installer** To simplify the installation process, Penguins-eggs provides its own system installer called krill. This installer is particularly useful when a GUI (Graphical User Interface) is not available, allowing for installation in various situations. However, if you are using a desktop system, Penguins-eggs recommends and configures the calamares GUI installer, ensuring a seamless and user-friendly experience. Penguins Eggs Linux spearheads a transformative revolution in the realm of system installation with the incorporation of its TUI (Text-based User Interface) / GUI (Graphical User Interface) installer, setting a new standard of versatility and accessibility within the landscape of Linux distributions.
6. **Repository Lists** One of the key advantages of Penguins-eggs is its commitment to utilizing only the original distro's packages. This means that no modifications are made to your repository lists, ensuring a safe and reliable environment. Penguins-eggs prioritizes maintaining the integrity and authenticity of your Linux distribution.
7. **Wardrobe** To enhance customization options, Penguins-eggs introduces the concept of Wardrobe. With Wardrobe and its various components, such as costumes, you can easily organize and manage your customizations, samples, and more. This feature enables a streamlined and efficient workflow, allowing you to tailor your Linux system to your preferences.

8. Supporting Multiple Distributions Eggs supporting multiple distributions and their derivatives Supports: Alpine, Arch, Debian, Fedora, Devuan, Manjaro, Ubuntu, and major derivatives like: Linuxmint, KDE neon, EndeavourOS, Garuda, etc. You can easily add more derivatives.
9. Supports Hardware Architectures supports a wide range of hardware architectures. Supports: i386, amd64 and arm64 architecture, from old PCs, and common PCs to single board computers like Raspberry Pi 4/5
10. Supports privacy and security Safe: only use the original distro's packages, without any modification in your repository lists. Penguins Eggs Linux embarks on a steadfast commitment to user security and system integrity through its default practice of exclusively utilizing original distributions' packages without any modifications in the repository lists. This resolute dedication to maintaining the pristine authenticity of packages reinforces Penguins Eggs' fundamental ethos of safety and reliability, fostering an environment characterized by unwavering trust in the integrity of the software ecosystem.

Wardrobe, Themes, and Addons

In April 2022, the wardrobe command was introduced to eggs. This addition serves as a comprehensive tool to assist and streamline the process of creating a customized version of Linux, starting from a command-line interface (CLI) system. I have embraced wardrobe for all my editions to enhance convenience, enabling me to better organize, consolidate, and manage my work effectively. To add a unique touch to my customizations, I have assigned bird names to each edition. Except for the "naked" edition, there are various options available, including "Colibri," "eagle," "duck," "owl," and "chicks" under the bookworm and plastilinux distributions. [bookworm](#) and [plastilinux](#). Furthermore, under Waydroid on the eggs' SourceForge page, you can find "wagtail" and "warbier." I have high hopes that people will take an interest in wardrobe and consider forking the main repository to incorporate their own customizations. By collaborating, we can achieve significant progress that would be challenging for a single developer to accomplish. If you would like to delve deeper into the wardrobe, I recommend reading the [Penguins Eggs' blog](#). post titled Wardrobe: Colibri, Duck, Eagle, and Owl, which provides further insights into its features and benefits. Furthermore, addons, predominantly themes, have been organized under the vendor's folder in the penguin's wardrobe. I encourage utilizing your wardrobe for all your customization needs to maintain consistency and organization throughout your work.

Note

For detailed instructions on using a wardrobe, please consult the wardrobe users' guide [wardrobe users' guide](#).

System Cloning and Data Handling

When creating a live distribution of your system using eggs produce, you must decide how to handle user data. You have three primary approaches: Default (Clean), Direct Clone, and Encrypted Clone.

1. Default Mode (Clean System)

Command: `eggs produce`

This is the default mode. It completely removes all user data and personal settings from the live distribution.

Best for: Creating a distributable ISO to share with others or creating a clean "factory reset" image.

Privacy: Maximum. No private data remains in the live system.

2. Clone (Unencrypted)

Command: `eggs produce --clone`

This command creates an exact replica of your current system, saving both system files and user data directly into the ISO.

Behavior: If someone obtains this ISO, they can boot it and immediately access your files, browsing history, and settings without a password.

Best for: Personal backups where the physical media (USB/DVD) remains in your secure possession.

Privacy Warning: Low. Data is not encrypted. Do not share this ISO publicly if it contains sensitive information.

3. Encrypted Clone (Secure Backup)

Commands: `eggs produce --homeencrypt` / `eggs produce --fullcrypt`

These options save your data within the ISO but protect it inside a LUKS (Linux Unified Key Setup) volume.

Behavior: Your user data is included but is not visible or accessible in the live system session. Even if a third party obtains the ISO, they cannot access the data without the LUKS passphrase.

Best for: Secure backups or transporting your environment across different machines.

■ [NOTE] Availability and Scope

`eggs produce --homecrypt`: Available for most distributions using either native penguins-eggs packages or the ApplImage. It protects sensitive data under `/home` and system accounts.

`eggs produce --fullcrypt`: Supported only on Debian Trixie and Devuan Excalibur, and exclusively when using native eggs packages. Unlike `homecrypt`, this option does not limit protection to `/home`, but encrypts the entire system.

Command	Data Included?	Encryption	Visible on Live System?	Use Case
<code>eggs produce</code>	No	N/A	No	Public Distribution / Clean Install
<code>eggs produce --clone</code>	Yes	No	Yes	Personal Unsecured Backup
<code>eggs produce --homecrypt</code>	Yes (/home)	Yes (LUKS)	Yes	Secure Backup of User Data
<code>eggs produce --fullcrypt</code>	Yes (Full)	Yes (LUKS)	Yes	Maximum Security Backup (Debian/Devuan only)

Calamares and krill

Calamares and Krill are powerful tools in the Eggs project, [calamares](#), offering versatile installation options for Linux systems. The Eggs project was specifically designed to utilize Calamares as the default system installer, providing users with the flexibility to customize their installations using themes. However, Eggs goes beyond Calamares by introducing its own installer called Krill, which focuses on command-line interface (CLI) installations, particularly for server environments.

Krill adopts a TUI interface that closely resembles Calamares, ensuring a consistent user experience. Leveraging the same configuration files created by Eggs for Calamares, Krill maintains compatibility and allows for seamless transitions between desktop and server installations. By simply adding the flag during installation, Krill enables unattended installations, streamlining the process for system administrators. Fine-tuning installation parameters becomes effortless as the configuration values can be modified in the `/etc/penguins-eggs.d/krill.yaml` file, facilitating automated deployments.

Thanks to the Eggs project's integration of Calamares and the introduction of Krill, users can enjoy a comprehensive installation toolkit. Whether one prefers the graphical interface of Calamares or the command-line efficiency of Krill, Eggs caters to diverse installation needs, making Linux setup a breeze.

Cuckoo

Just like the cuckoo bird lays its eggs in the nests of other birds, the Eggs project introduces a similar concept in the form of a self-configuring PXE service. This service allows you to boot and install your ISO on networked computers that are not originally configured for your specific ISO. With the command `cuckoo` you can deploy a newly created ISO on an already installed system, or you can live to boot the ISO itself. This means that you can either install your ISO on existing systems or directly run the ISO without the need for a permanent installation.

Tip

By leveraging the `cuckoo` command, the Eggs project provides a convenient method for deploying and testing your ISO on a variety of networked computers, expanding the possibilities for system installations and evaluations.

Mom and Dad

I have introduced two helpful built-in assistants: Mom and Dad. Mom, based on the `easybashgui` [easybashgui](#) script, serves as a comprehensive guide, providing explanations of various commands and documentation. This ensures that users have access to clear instructions and information as they navigate through Eggs' functionalities. On the other hand, Dad serves as a convenient shortcut for properly configuring Eggs. By simply typing `sudo eggs dad` and following the straightforward instructions, users can quickly configure Eggs to meet their specific requirements. For even faster configuration, utilizing the command `sudo eggs dad -d` allows for a complete reset of the configuration, loading default settings, and deleting any created ISOs. Once Eggs is properly configured, generating your live environment becomes a breeze. Just type `sudo eggs produce` to effortlessly generate your live ISO. With this streamlined workflow, Eggs empowers users to efficiently create customized live environments tailored to their needs. Whether you rely on Mom's guidance or Dad's configuration shortcuts, Eggs offers a user-friendly experience for ISO creation and customization.

Yolk

Yolk is a local repository that is bundled within the LiveCD of Eggs. This repository contains a carefully curated selection of essential packages required for installation. Yolk serves as a valuable resource, as it allows you to install your

system confidently, even without an active internet connection. By including Yolk in the LiveCD, Eggs ensures that all the necessary packages are readily available during the installation process. This eliminates the dependency on an internet connection, making it possible to install your system in offline environments or situations where internet access is limited or unavailable. Yolk acts as a safety net, providing the minimum set of indispensable packages required for a successful installation. This guarantees a smooth and reliable installation experience, regardless of the availability of an internet connection. With Yolk by your side, you can confidently proceed with system installations, knowing that the essential packages are at your disposal.

Packages

Eggs offers support for a variety of packages. Specifically, for Debian, Devuan, and Ubuntu, Eggs utilizes .deb packages that are compatible with both amd_64 and i386 architectures. This ensures seamless integration with these distributions, allowing users to easily install and utilize Eggs' features. On the other hand, Arch and ManjaroLinux have their own packaging system known as PKGBUILDs. Eggs is designed to work harmoniously with these distributions, leveraging the specific packaging structure provided by PKGBUILDs. This ensures that Eggs can seamlessly integrate into Arch and ManjaroLinux environments, providing users with a consistent and optimized experience. By adapting to the packaging systems used by different distributions, Eggs ensures compatibility and ease of use across a wide range of Linux environments. Whether you're using Debian, Devuan, Ubuntu, Arch, or ManjaroLinux, Eggs is equipped to support your preferred distribution, enabling you to make the most of its features and functionalities.

Debian families

Eggs caters to the Debian family of distributions, offering a seamless installation experience through deb packages.

These deb packages are available for multiple architectures, including amd64, i386, and arm64.

The availability of penguins-eggs as a deb package simplifies the installation process for users of Debian-based distributions. Whether you are running a 64-bit (amd64) or 32-bit (i386) architecture, or even an arm64 architecture, Eggs has you covered. This ensures that users across a wide range of Debian-based systems can easily download, install, and utilize Eggs' features.

By providing deb packages for various architectures, Eggs promotes accessibility and inclusivity, allowing users on different hardware platforms to benefit from its functionality. Whether you're using a traditional desktop computer or an ARM-

based device, Eggs ensures compatibility and a consistent experience across the Debian family of distributions.

The packages can be installed on Debian, Devuan, or Ubuntu-based distributions without the need to worry about the specific version. Whether you're using Buster, Bullseye, Bookworm, Trixie, Chimaera, Daedalus, Focal, Jammy or Noble, eggs is reported to work across these versions. However, it's important to ensure compatibility with the respective processor architecture.

Ubuntu bionic cannot install nodejs >16, so I packaged a specific version for bionic, still largely used.

The packages penguins-eggs provided include standard scripts for preinst, postinst, prerm, and postrm. These scripts play a crucial role in the installation and management of the packages. The preinst script is executed before the package is installed, allowing for any necessary preparations or configurations. The postinst script is executed after the package installation, enabling additional setup or customization. Similarly, the prerm script is executed before the package is removed, while the postrm script is executed after the package removal.

In addition to the scripts, penguins-eggs package also include man pages, bash-completion and other.

man pages serve as documentation for the installed packages, providing detailed information on their usage, configuration options, and other relevant details. The inclusion of man pages ensures that users have access to comprehensive documentation, enabling them to effectively utilize and manage the Eggs packages.

Overall, Eggs' packages offer a comprehensive and user-friendly experience, with standard scripts and detailed documentation, making installation and management hassle-free on Debian, Devuan, and Ubuntu-based distributions.

Install eggs

There are multiple methods available, lately I prefer fresh-eggs way, working for all the supported distros: Arch, Debian, Devuan, Manjaro, Ubuntu and derivatives.:

```
git clone https://github.com/pieroproietti/fresh-eggs
cd fresh-eggs
sudo ./fresh-eggs.sh
```

fresh-eggs on Debian families add automatically the penguins-eggs-ppa repository and, when need, add nodesource repository. On Arch, chaotic-aur repo will be added.

This let to install, update, remove penguins-eggs like a native package.

Debian families

`penguins-eggs-25.x.x` depend on `nodejs >18`, not directly available in all the distros. We can rely on [nodesource](#) adding them.

Download and install penguins-eggs packages

To install Eggs, the simplest method is to download the package `penguins-eggs` from the project's SourceForge page [package eggs](#) and install it on your system.

After downloading the appropriate package, based on your system's architecture, you can proceed with the installation. If you are using an amd64 system, run the following command in the terminal:

```
sudo dpkg -i penguins_eggs_25.x.x-1_amd64.deb
```

For i386 systems, the command would be: `sudo dpkg -i penguins_eggs_25.x.x-1_i386.deb`

Executing these commands will initiate the installation process and install Eggs on your system. Once Eggs is successfully installed, you have the option to enhance its functionality by adding the `penguins-eggs-ppa` repository. This repository provides additional tools and features for Eggs. To add the `penguins-eggs-ppa` repository, run the following command in the terminal: `sudo eggs tools ppa --install` This command will add the `penguins-eggs-ppa` repository to your system, allowing you to access updated versions of Eggs and additional tools provided by the repository. By following these steps, you can easily install Eggs, add the `penguins-eggs-ppa` repository, [penguins-eggs-ppa](#), and unlock further capabilities and enhancements for your Eggs installation.

Using penguins-eggs-ppa

For derivatives of Debian, Devuan, and Ubuntu, such as Linuxmint, LMDE, etc., `fresh-eggs` will typically work as well. However, if needed, you can manually add the `penguins-eggs-ppa` repository by copying and pasting the following two lines into a terminal:

```
curl -fsSL https://pieroproietti.github.io/penguins-eggs-ppa/KEY.gpg  
echo "deb [arch=$(dpkg --print-architecture)] https://pieroproietti."
```

After adding the repository, update your package repositories and install Eggs by running the following command:

```
sudo apt update && sudo apt install penguins_eggs
```

Executing these commands will update your package sources and install Eggs on your system.

Upgrade eggs

To upgrade Eggs, the process will vary depending on whether you are using the penguins-eggs-ppa repository or not. Here's how you can upgrade Eggs with both approaches: If you have already added the penguins-eggs-ppa repository, you can upgrade Eggs alongside other packages on your system by running the following command:

```
sudo apt upgrade
```

Tip

This command will check for updates for all installed packages, including Eggs, and upgrade them to their latest versions if available.

Note

On the other hand, if you have not added the penguins-eggs-ppa repository, you can manually upgrade Eggs by downloading the new version from the SourceForge page [here](#). Once you have downloaded the appropriate package for your system architecture, follow these steps:

1. Install the package using the gdebi command (assuming you have gdebi installed):

```
sudo gdebi penguins_eggs_25.x.x-1_amd64.deb
```

or for i386 systems:

```
sudo dpkg -i penguins_eggs_25.x.x-1_i386.deb
```

2. In case of any missing dependencies, you can resolve them by running the following command:

```
sudo apt install -f
```

This will automatically install any required dependencies for Eggs.

Tip

By following these instructions, you can upgrade Eggs either through the penguins-eggs-ppa repository or by manually downloading and installing the latest version from the SourceForge page. Ensure that you choose the appropriate method based on your current setup to keep Eggs up to date with the latest enhancements and bug fixes.

Arch families

Eggs has been available in the Arch User Repository (AUR) for quite some time, thanks to the support of the Arch Linux community. Although I was initially unaware of its presence, I am now directly maintaining the AUR version of [penguins-eggs](#). Additionally, I am actively participating in the Manjaro Community Repository, specifically for the [penguins-eggs](#) package.

Being present in the AUR signifies that Eggs is available for Arch Linux users to easily install and manage through their package managers. The AUR is a community-driven repository that allows users to contribute and maintain packages that are not officially supported by Arch Linux. By maintaining the AUR version of penguins-eggs, I can ensure that Arch Linux users have access to the latest updates and improvements for Eggs.

Tip

Furthermore, my participation in the Manjaro Community Repository demonstrates my commitment to providing support for Eggs on the Manjaro distribution. Manjaro is a popular Arch-based Linux distribution known for its user-friendly approach and community-driven development. By actively contributing to the Manjaro Community Repository, I can ensure that Eggs remains compatible and well-integrated with the Manjaro ecosystem.

Tip

In summary, Eggs is available in the AUR and is directly maintained by me. Additionally, I am actively involved in the Manjaro Community Repository to provide support for Eggs on the Manjaro distribution. This ensures that users of Arch Linux and its derivatives, such as Manjaro, can easily access and benefit from using Eggs in their systems.

Arch

To install penguins-eggs on Arch Linux, there are multiple methods available. One option is to install it directly from the Arch User Repository (AUR) by adding the chaotic-AUR repository. Here's how you can do it:

1. Add the Chaotic-AUR repository to your system. You can find the repository at <https://aur.chaotic.cx/>.

2. After adding the Chaotic-AUR repository, open a terminal and run the following command to install penguins-eggs using pacman:

```
sudo pacman -Sy penguins-eggs
```

This command will synchronize the package databases and install penguins-eggs on your system.

Alternatively, you can use a utility called `fresh-eggs`. This script will add the AUR repository and install penguins-eggs on your system.

It's possible too to use the popular AUR helper tool called `yay`. Simply run the following command:

```
yay penguins-eggs
```

`yay` will handle the installation process for you, including any necessary dependencies.

If you prefer to build from source, you can download the sources from the AUR repository. Here are the steps:

1. Clone the penguins-eggs repository from the AUR:

```
git clone https://aur.archlinux.org/packages/penguins-eggs
```

2. Change to the penguins-eggs directory:

```
cd penguins-eggs
```

3. Build and install the package using `makepkg`:

```
makepkg -srcCi
```

Tip

This command will compile the source code, create a package, and install it on your system. These methods provide various ways to install penguins-eggs on Arch Linux, allowing you to choose the one that suits your preferences and workflow.

Starting from version 9.4.3, penguins-eggs is now included in the Manjaro community repository, making it even easier to install on Manjaro Linux. To install penguins-eggs on Manjaro, you can use the pamac package manager with the following command:

```
pamac install penguins-eggs
```

This command will fetch the package from the Manjaro community repository and install it on your system.

Alternatively, if you prefer to manually manage the installation process, you can clone the penguins-eggs package from the Manjaro community repository and build it from source. Here are the steps:

1. Clone the penguins-eggs package from the Manjaro community repository:

```
git clone https://gitlab.manjaro.org/packages/community/penguins-eggs
```

2. Change to the penguins-eggs directory:

```
cd penguins-eggs
```

3. Build and install the package using makepkg:

```
makepkg -srcCi
```

Tip

This command will compile the source code, create a package, and install it on your system.

By including penguins-eggs in the Manjaro community repository, Manjaro Linux users can easily access and install the package using their preferred package manager. The Manjaro community repository is specifically dedicated to packages that are supported by the Manjaro community, ensuring that penguins-eggs is well-integrated and compatible with the Manjaro distribution.

Tip

Whether you choose to install penguins-eggs using pamac or by manually building it from source, you can enjoy the benefits of this package on your Manjaro Linux system.

Usage

Once the package has been installed, you can have the new eggs command. Typing eggs will get the list of commands, and typing eggs produce --help will get the eggs produce command help screen. You can also use the command autocomplete with the TABS key, you will get the possible choices for each command. In addition, there is a man page, so by typing man eggs you will get that help as well. You can also use the eggs mom command that interactively allows you to consult the help for all commands and online documentation.

Examples

Here are some examples of how to use penguins-eggs to create live systems with different configurations:

1. To create a live system without user data, run the following command with sudo:

```
sudo eggs produce
```

This command will generate a live system without any user data included.

2. To create a live system with user data that is not encrypted, use the --clone flag:

```
sudo eggs produce --clone
```

This command will produce a live system that includes user data without encryption.

3. If you want to create a live system with encrypted user data, use the --cryptedclone flag:

```
sudo eggs produce --cryptedclone
```

This command will generate a live system with encrypted user data.

Compression

By default, penguins-eggs uses fasted compression zstd level 3 for efficiency during the creation process. However, if you want a more compressed ISO file, you can choose --pendrive flag zstd level 15 optimized for pendrives, --standard

flag, use xz, or --max flag using xz -Xbcj to get the maximum level of compression. For example:

```
sudo eggs produce
sudo eggs produce --pendrive
sudo eggs produce --standard
sudo eggs produce --max
```

Tip

This command will apply different compression to the ISO file, resulting in a smaller file size or in a longer process. Consult [Penguins Eggs' official guide](#) for more detailed information.

Commands

- `eggs adapt`
- `eggs autocomplete [SHELL]`
- `eggs calamares`
- `eggs config`
- `eggs cuckoo`
- `eggs dad`
- `eggs export appimage`
- `eggs export iso`
- `eggs export pkg`
- `eggs export tarballs`
- `eggs help [COMMAND]`
- `eggs kill`
- `eggs krill`
- `eggs love`
- `eggs mom`
- `eggs produce`
- `eggs setup install`
- `eggs setup purge`
- `eggs status`
- `eggs tools clean`
- `eggs tools repo`
- `eggs tools skel`
- `eggs tools stat`
- `eggs tools yolk`
- `eggs update`
- `eggs version`
- `eggs wardrobe get [REPO]`

- [eggs wardrobe list \[REP0\]](#)
- [eggs wardrobe show \[REP0\]](#)
- [eggs wardrobe wear \[REP0\]](#)

eggs adapt

adapt monitor resolution for VM only

```
USAGE
$ eggs adapt [-h] [-v]

FLAGS
-h, --help      Show CLI help.
-v, --verbose

DESCRIPTION
adapt monitor resolution for VM only

EXAMPLES
$ eggs adapt
```

See code: [src/commands/adapt.ts](#)

eggs autocomplete [SHELL]

Display autocomplete installation instructions.

```
USAGE
$ eggs autocomplete [SHELL] [-r]

ARGUMENTS
[SHELL]  (zsh|bash|powershell) Shell type

FLAGS
-r, --refresh-cache  Refresh cache (ignores displaying instructions)

DESCRIPTION
Display autocomplete installation instructions.

EXAMPLES
$ eggs autocomplete

$ eggs autocomplete bash
```

```
$ eggs autocomplete zsh

$ eggs autocomplete powershell

$ eggs autocomplete --refresh-cache
```

See code: [@oclif/plugin-autocomplete](#)

eggs calamares

a GUI system installer - install and configure calamares

USAGE

```
$ eggs calamares [-h] [-i] [-n] [-p] [-r] [--remove] [--theme <val
```

FLAGS

-h, --help	Show CLI help.
-i, --install	install calamares and its dependencies
-n, --nointeractive	no user interaction
-p, --policies	configure calamares policies
-r, --release	release: remove calamares and all its depende
-v, --verbose	
--remove	remove calamares and its dependencies
--theme=<value>	theme/branding for eggs and calamares

DESCRIPTION

a GUI system installer - install and configure calamares

EXAMPLES

```
sudo eggs calamares

sudo eggs calamares --install

sudo eggs calamares --install --theme=/path/to/theme

sudo eggs calamares --remove
```

See code: [src/commands/calamares.ts](#)

eggs config

Configure eggs to run it

USAGE

```
$ eggs config [-c] [-h] [-n] [-v]
```

FLAGS

-c, --clean	remove old configuration before to create new
-h, --help	Show CLI help.
-n, --nointeractive	no user interaction
-v, --verbose	verbose

DESCRIPTION

Configure eggs to run it

EXAMPLES

```
sudo eggs config
```

```
sudo eggs config --clean
```

```
sudo eggs config --clean --nointeractive
```

See code: [src/commands/config.ts](#)

eggs cuckoo

PXE start with proxy-dhcp

USAGE

```
$ eggs cuckoo [-h] [-v]
```

FLAGS

-h, --help	Show CLI help.
-v, --verbose	verbose

DESCRIPTION

PXE start with proxy-dhcp

EXAMPLES

```
sudo eggs cuckoo
```

See code: [src/commands/cuckoo.ts](#)

eggs dad

ask help from daddy - TUI configuration helper

USAGE

```
$ eggs dad [-c] [-d] [-f <value>] [-n] [-h] [-v]
```

FLAGS

```
-c, --clean          remove old configuration before to create  
-d, --default        reset to default values  
-f, --file=<value>  use a file configuration custom  
-h, --help           Show CLI help.  
-n, --nointeractive  no user interaction  
-v, --verbose
```

DESCRIPTION

ask help from daddy - TUI configuration helper

EXAMPLES

```
sudo dad
```

```
sudo dad --clean
```

```
sudo dad --default
```

See code: <src/commands/dad.ts>

eggs export appimage

export penguins-eggs AppImage to the destination host

USAGE

```
$ eggs export appimage [-c] [-h] [-v]
```

FLAGS

```
-c, --clean          remove old .AppImage before to copy  
-h, --help           Show CLI help.  
-v, --verbose        verbose
```

DESCRIPTION

export penguins-eggs AppImage to the destination host

EXAMPLES

```
$ eggs export pkg
```

```
$ eggs export pkg --clean
```

```
$ eggs export pkg --all
```

See code: [src/commands/export/appimage.ts](#)

eggs export iso

export remastered ISO in the destination host

USAGE

```
$ eggs export iso [-C] [-c] [-h] [-v]
```

FLAGS

```
-C, --checksum  export checksums md5 and sha256
-c, --clean     delete old ISOs before to copy
-h, --help      Show CLI help.
-v, --verbose   verbose
```

DESCRIPTION

export remastered ISO in the destination host

EXAMPLES

```
$ eggs export iso
```

```
$ eggs export iso --clean
```

See code: [src/commands/export/iso.ts](#)

eggs export pkg

export penguins-eggs package to the destination host

USAGE

```
$ eggs export pkg [-a] [-c] [-h] [-v]
```

FLAGS

```
-a, --all      export all archs
-c, --clean     remove old .deb before to copy
-h, --help      Show CLI help.
-v, --verbose   verbose
```

DESCRIPTION

export penguins-eggs package to the destination host

EXAMPLES

```
$ eggs export pkg

$ eggs export pkg --clean

$ eggs export pkg --all
```

See code: [src/commands/export/pkg.ts](#)

eggs export tarballs

export pkg/iso/tarballs to the destination host

USAGE

```
$ eggs export tarballs [-c] [-h] [-v]
```

FLAGS

```
-c, --clean    remove old .deb before to copy
-h, --help     Show CLI help.
-v, --verbose  verbose
```

DESCRIPTION

```
export pkg/iso/tarballs to the destination host
```

EXAMPLES

```
$ eggs export tarballs

$ eggs export tarballs --clean
```

See code: [src/commands/export/tarballs.ts](#)

eggs help [COMMAND]

Display help for eggs.

USAGE

```
$ eggs help [COMMAND...] [-n]
```

ARGUMENTS

```
[COMMAND...]  Command to show help for.
```

FLAGS

```
-n, --nested-commands  Include all nested commands in the output.
```

DESCRIPTION

Display help for eggs.

See code: [@oclif/plugin-help](#)

eggs kill

kill the eggs/free the nest

USAGE

```
$ eggs kill [-h] [-i] [-n] [-v]
```

FLAGS

-h, --help	Show CLI help.
-i, --isos	erase all ISOs on remote mount
-n, --nointeractive	no user interaction
-v, --verbose	verbose

DESCRIPTION

kill the eggs/free the nest

EXAMPLES

```
sudo eggs kill
```

See code: [src/commands/kill.ts](#)

eggs krill

a TUI system installer - install the system

USAGE

```
$ eggs krill [-b] [-c] [-k] [-d <value>] [-H] [-h] [-i] [-n] [-N] [-u] [-v]
```

FLAGS

-H, --halt	Halt the system after installation
-N, --none	Swap none: 256M
-R, --replace=<value>	Replace partition. eg: --replace /dev/sda3
-S, --suspend	Swap suspend: RAM x 2
-b, --btrfs	Format btrfs
-c, --chroot	chroot before to end
-d, --domain=<value>	Domain name, default: .local
-h, --help	Show CLI help.
-i, --ip	hostname as ip, eg: ip-192-168-1-33

-k, --crypted	Crypted CLI installation
-n, --nointeractive	no user interaction
-p, --pve	Proxmox VE install
-r, --random	Add random to hostname, eg: colibri-ay412dt
-s, --small	Swap small: RAM
-t, --testing	Just testing krill
-u, --unattended	Unattended installation
-v, --verbose	Verbose

DESCRIPTION

a TUI system installer - install the system

EXAMPLES

```
sudo eggs install
```

```
sudo eggs install --unattended --halt
```

```
sudo eggs install --chroot
```

See code: [src/commands/krill.ts](#)

eggs love

the simplest way to get an egg!

USAGE

```
$ eggs love [-h] [-v] [-H] [-n] [-c] [-k] [-f]
```

FLAGS

-H, --hidden	stealth mode
-c, --clone	clone (uncrypted)
-f, --fullcrypt	clone crypted full
-h, --help	Show CLI help.
-k, --homecrypt	clone crypted home
-n, --nointeractive	no user interaction
-v, --verbose	

DESCRIPTION

the simplest way to get an egg!

EXAMPLES

```
$ eggs auto
```

See code: [src/commands/love.ts](#)

eggs mom

ask help from mommy - TUI helper

USAGE

```
$ eggs mom [-h]
```

FLAGS

-h, --help Show CLI help.

DESCRIPTION

ask help from mommy - TUI helper

EXAMPLES

```
$ eggs mom
```

See code: <src/commands/mom.ts>

eggs produce

produce a live image from your system

USAGE

```
$ eggs produce [--addons <value>...] [--basename <value>] [-c] [-k  
[-K <value>] [--links <value>...] [-m] [-N] [-n] [-p] [-P <value>  
[-y]
```

FLAGS

-H, --hidden	stealth mode
-K, --kernel=<value>	kernel version
-N, --noicon	no icon eggs on desktop
-P, --prefix=<value>	prefix
-S, --standard	standard compression: xz -b 1M
-c, --clone	clone (uncrypted)
-f, --fullcrypt	clone crypted full
-h, --help	Show CLI help.
-i, --includeRootHome	folder /root is included on live
-k, --homecrypt	clone crypted home
-m, --max	max compression: xz -Xbcj ...
-n, --nointeractive	no user interaction
-p, --pendrive	optimized for pendrive: zstd -b 1M -Xcc
-s, --script	script mode. Generate scripts to manage
-v, --verbose	verbose
-y, --yolk	force yolk renew

```
--addons=<value>...    addons to be used: adapt, pve, rsupport
--basename=<value>      basename
--excludes=<value>...   use: static, homes, home
--links=<value>...      desktop links
--release               release: remove penguins-eggs, calamare
--theme=<value>         theme for livecd, calamares branding ar
```

DESCRIPTION

produce a live image from your system

EXAMPLES

```
sudo eggs produce          # zstd fast compression

sudo eggs produce --pendrive    # zstd compression optimized

sudo eggs produce --clone      # clear clone (unencrypted)

sudo eggs produce --homecrypt   # clone crypted home (all insid

sudo eggs produce --fullcrypt   # clone crypted full (entire sy

sudo eggs produce --basename=colibri
```

See code: [src/commands/produce.ts](#)

eggs setup install

Automatically check and install system prerequisites

USAGE

```
$ eggs setup install
```

DESCRIPTION

Automatically check and install system prerequisites

EXAMPLES

```
$ eggs setup          # this help

sudo eggs setup install    # install native dependencies

sudo eggs setup purge      # purge all configurations, a
```

See code: [src/commands/setup/install.ts](#)

eggs setup purge

Automatically check and install system prerequisites

USAGE

```
$ eggs setup purge
```

DESCRIPTION

Automatically check and install system prerequisites

EXAMPLES

```
$ eggs setup                                # this help

sudo eggs setup install                    # install native dependencies

sudo eggs setup purge                      # purge all configurations, a
```

See code: [src/commands/setup/purge.ts](#)

eggs status

informations about eggs status

USAGE

```
$ eggs status [-h] [-v]
```

FLAGS

```
-h, --help      Show CLI help.
-v, --verbose
```

DESCRIPTION

informations about eggs status

EXAMPLES

```
$ eggs status
```

See code: [src/commands/status.ts](#)

eggs tools clean

clean system log, apt, etc

USAGE

```
$ eggs tools clean [-h] [-n] [-v]
```

FLAGS

-h, --help	Show CLI help.
-n, --nointeractive	no user interaction
-v, --verbose	verbose

DESCRIPTION

clean system log, apt, etc

EXAMPLES

```
sudo eggs tools clean
```

See code: <src/commands/tools/clean.ts>

eggs tools repo

add/remove penguins-repos

USAGE

```
$ eggs tools repo [-a] [-h] [-n] [-r] [-v]
```

FLAGS

-a, --add	add penguins-repos
-h, --help	Show CLI help.
-n, --nointeractive	no user interaction
-r, --remove	remove penguins-repos
-v, --verbose	verbose

DESCRIPTION

add/remove penguins-repos

EXAMPLES

```
sudo eggs tools repo --add
```

```
sudo eggs tools repo --remove
```

See code: <src/commands/tools/repo.ts>

eggs tools skel

update skel from home configuration

USAGE

```
$ eggs tools skel [-h] [-u <value>] [-v]
```

FLAGS

```
-h, --help          Show CLI help.  
-u, --user=<value>  user to be used  
-v, --verbose
```

DESCRIPTION

```
update skel from home configuration
```

EXAMPLES

```
sudo eggs tools skel
```

```
sudo eggs tools skel --user user-to-be-copied
```

See code: <src/commands/tools/skel.ts>

eggs tools stat

get statistics from sourceforge

USAGE

```
$ eggs tools stat [-h] [-m] [-y]
```

FLAGS

```
-h, --help    Show CLI help.  
-m, --month   current month  
-y, --year    current year
```

DESCRIPTION

```
get statistics from sourceforge
```

EXAMPLES

```
$ eggs tools stat
```

```
$ eggs tools stat --month
```

```
$ eggs tools stat --year
```

See code: <src/commands/tools/stat.ts>

eggs tools yolk

configure eggs to install without internet

USAGE

```
$ eggs tools yolk [-h] [-v]
```

FLAGS

```
-h, --help      Show CLI help.  
-v, --verbose
```

DESCRIPTION

```
configure eggs to install without internet
```

EXAMPLES

```
sudo eggs tools yolk
```

See code: <src/commands/tools/yolk.ts>

eggs update

update the Penguins' eggs tool

USAGE

```
$ eggs update [-h] [-v]
```

FLAGS

```
-h, --help      Show CLI help.  
-v, --verbose   verbose
```

DESCRIPTION

```
update the Penguins' eggs tool
```

EXAMPLES

```
$ eggs update
```

See code: <src/commands/update.ts>

eggs version

USAGE

```
$ eggs version [--json] [--verbose]
```

FLAGS

```
--verbose  Show additional information about the CLI.
```

GLOBAL FLAGS

`--json` Format output as json.

FLAG DESCRIPTIONS

`--verbose` Show additional information about the CLI.

Additionally shows the architecture, node version, operating sys

See code: [@oclif/plugin-version](#)

eggs wardrobe get [REP0]

get warorobe

USAGE

```
$ eggs wardrobe get [REP0] [-h] [-v]
```

ARGUMENTS

[REP0] repository to get

FLAGS

`-h, --help` Show CLI help.

`-v, --verbose`

DESCRIPTION

get warorobe

EXAMPLES

```
$ eggs wardrobe get
```

```
$ eggs wardrobe get your-wardrobe
```

See code: [src/commands/wardrobe/get.ts](#)

eggs wardrobe list [REP0]

list costumes and accessoires in wardrobe

USAGE

```
$ eggs wardrobe list [REP0] [-d <value>] [-h] [-v]
```

ARGUMENTS

[REP0] wardrobe to get

FLAGS

-d, --distro=<value> distro
-h, --help Show CLI help.
-v, --verbose

DESCRIPTION

list costumes and accessoires in wardrobe

EXAMPLES

```
$ eggs wardrobe list  
  
$ eggs wardrobe list your-wardrobe  
  
$ eggs wardrobe list --distro arch
```

See code: <src/commands/wardrobe/list.ts>

eggs wardrobe show [REP0]

show costumes/accessories in wardrobe

USAGE

```
$ eggs wardrobe show [REP0] [-h] [-j] [-v] [-w <value>]
```

ARGUMENTS

[REP0] costume to show

FLAGS

-h, --help Show CLI help.
-j, --json output JSON
-v, --verbose
-w, --wardrobe=<value> wardrobe

DESCRIPTION

show costumes/accessories in wardrobe

EXAMPLES

```
$ eggs wardrobe show colibri  
  
$ eggs wardrobe show accessories/firmwares  
  
$ eggs wardrobe show accessories/
```

See code: <src/commands/wardrobe/show.ts>

eggs wardrobe wear [REP0]

wear costume/accessories from wardrobe

USAGE

```
$ eggs wardrobe wear [REP0] [-h] [-a] [-f] [-v] [-w <value>]
```

ARGUMENTS

[REP0] costume to wear

FLAGS

-a, --no_accessories	not install accessories
-f, --no_firmwares	not install firmwares
-h, --help	Show CLI help.
-v, --verbose	
-w, --wardrobe=<value>	wardrobe

DESCRIPTION

wear costume/accessories from wardrobe

EXAMPLES

```
sudo eggs wardrobe wear duck
```

```
sudo eggs wardrobe wear accessories/firmwares
```

```
sudo eggs wardrobe wear wagtail/waydroid
```

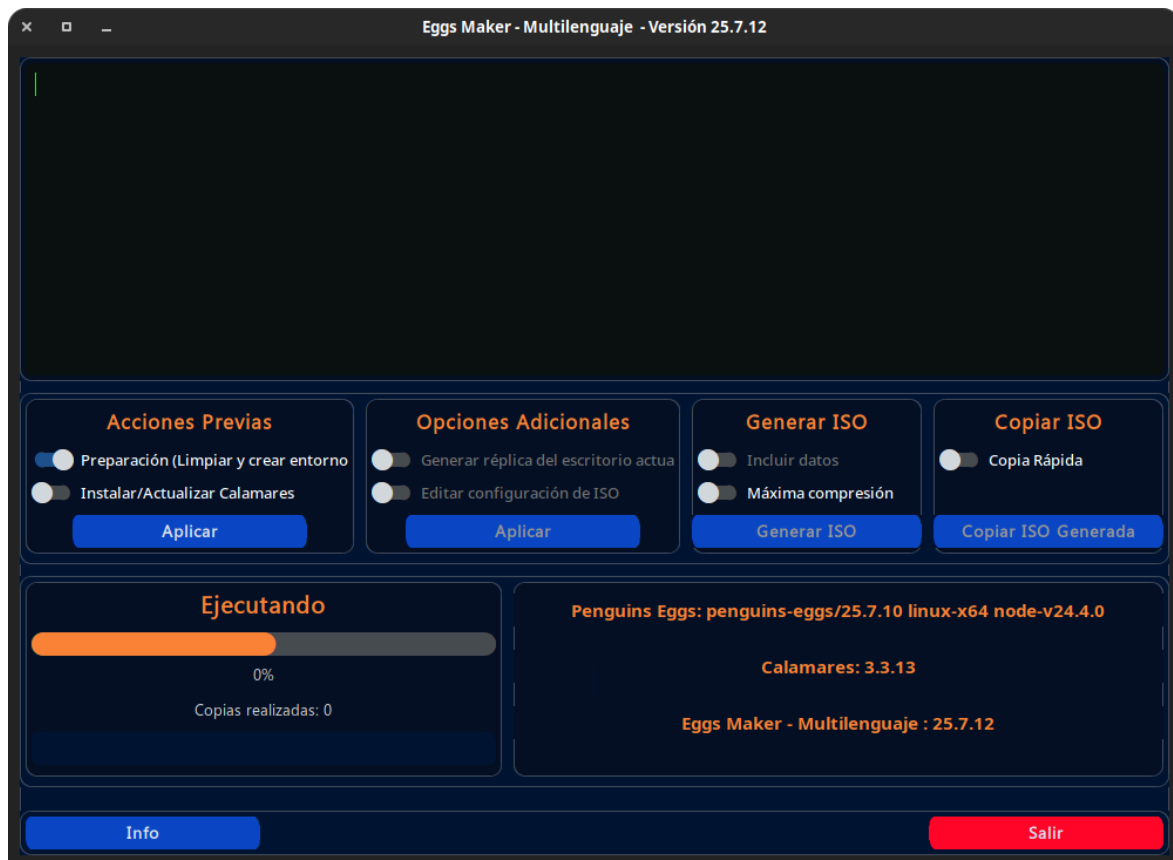
See code: <src/commands/wardrobe/wear.ts>

GUI

There are two GUIs for penguins-eggs at the moment: eggsmaker and penGUI.

eggsmaker

A project by [Jorge Luis Endres](#).



eggsmaker is a graphical interface for penguins-eggs.

Written by my friend Jorge Luis Endres, it is essential and functional. It doesn't cover all the possibilities of penguins-eggs, but in the end, a GUI should be simple and intuitive.

I like it, I hope you like it too, and I thank Jorge for his daring.

eggsmaker packages are available on [Jorge gdrive](#).

Book

My friend [Hosein Seilany](#) founder of [predator-os](#), has written a book on Penguins's eggs, with my participation. It's a remarkable work - even in size and weight - so it's a great honor to [announce](#) it here!

Penguin Eggs Tool

Create your own Linux distribution based easily

Hossein Seilani

Piero Proietti



Creating Linux distro with and ISO image file by using Penguins Eggs platform tool

First Edition

2024

That's all, Folks!

One of the standout features of Penguins Eggs' is its hassle-free setup. It comes with all the necessary configurations, making it a convenient choice for users. Just like in real life, the magic of Penguins Eggs' lies within - no additional setup required!

More Information

In addition to the official guide, there are other resources available for Penguins Eggs' users, particularly developers. These resources can be found in the [penguins-eggs repository](#) under the [documents](#) section.

Some noteworthy documents include:

- [Hens: Different Species](#): A brief guide on using Penguins Eggs' in Debian, Arch, and Manjaro.
- [Arch-naked](#): A blog post detailing how to create an Arch naked live, install it, and customize the resulting system into a graphics development station.

If you have any questions or need further assistance, feel free to contact me via email at pieroproietti@gmail.com. You can also stay updated by following my [blog](#) or connecting with me on , [Telegram](#), [Mastodom](#), [Facebook](#), [GitHub](#), [Jitsi](#), [Reddit](#) or [Twitter](#), [Mastodom](#).

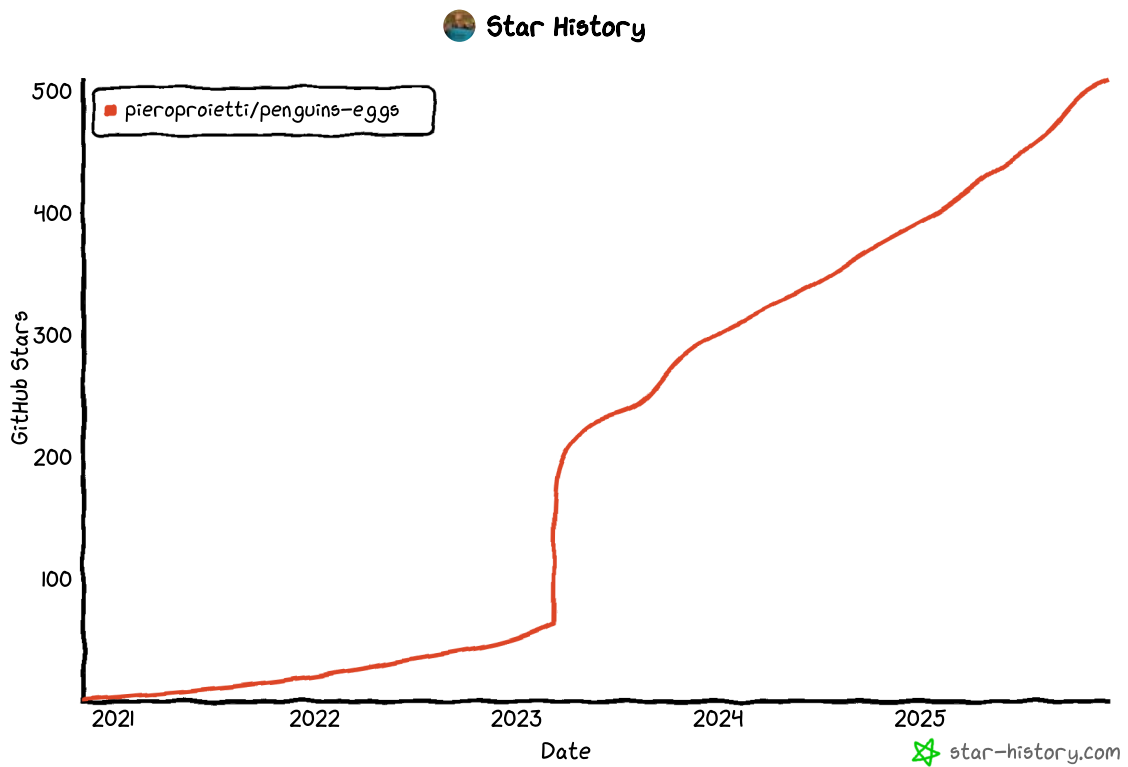
A word of thanks

- This README would not be so well cared for if not for the work of [Hosein Seilain](#) who spent his time revising and supplementing the text;
- The eggs icon was designed by [Charlie Martinez](#);
- and a word of thanks to all of you who are using it and providing feedback and motivation to continue it.

Thank you!

Star History

This project collects stars, look to the sky... contribute!



Copyright and licenses

Copyright (c) 2017, 2025 [Piero Proietti](#), dual licensed under the MIT or GPL Version 2 licenses.