



Open-source **Jazz Jackrabbit 2** reimplementation

Brought to you by [@deathkiller](#)

[Overview](#) [Release Notes](#) [Help](#) [Multiplayer](#)

Introduction

Jazz² Resurrection is reimplementation of the game **Jazz Jackrabbit 2** released in 1998. Supports various versions of the game (Shareware Demo, Holiday Hare '98, The Secret Files and Christmas Chronicles). Also, it partially supports some features of JJ2+ extension and MLLE. This project is hosted at [GitHub.com/deathkiller/jazz2-native](#).

Jazz² Resurrection supports various versions of the original game, but it is recommended to use *The Secret Files*.

build passing

release **v2.9.1**

downloads **44k**

code quality **A**

license **GPL-3.0**

chat **83 online**

Downloads

Latest version was released on **November 1st, 2024** (17 days ago). Release notes can be found [here](#).

Once you download and install *Android* version, you will be prompted with some security permissions to install app outside the Play Store.

Play on **Windows**

Linux

Play on **Android**

Play in **Browser**

macOS Switch Xbox

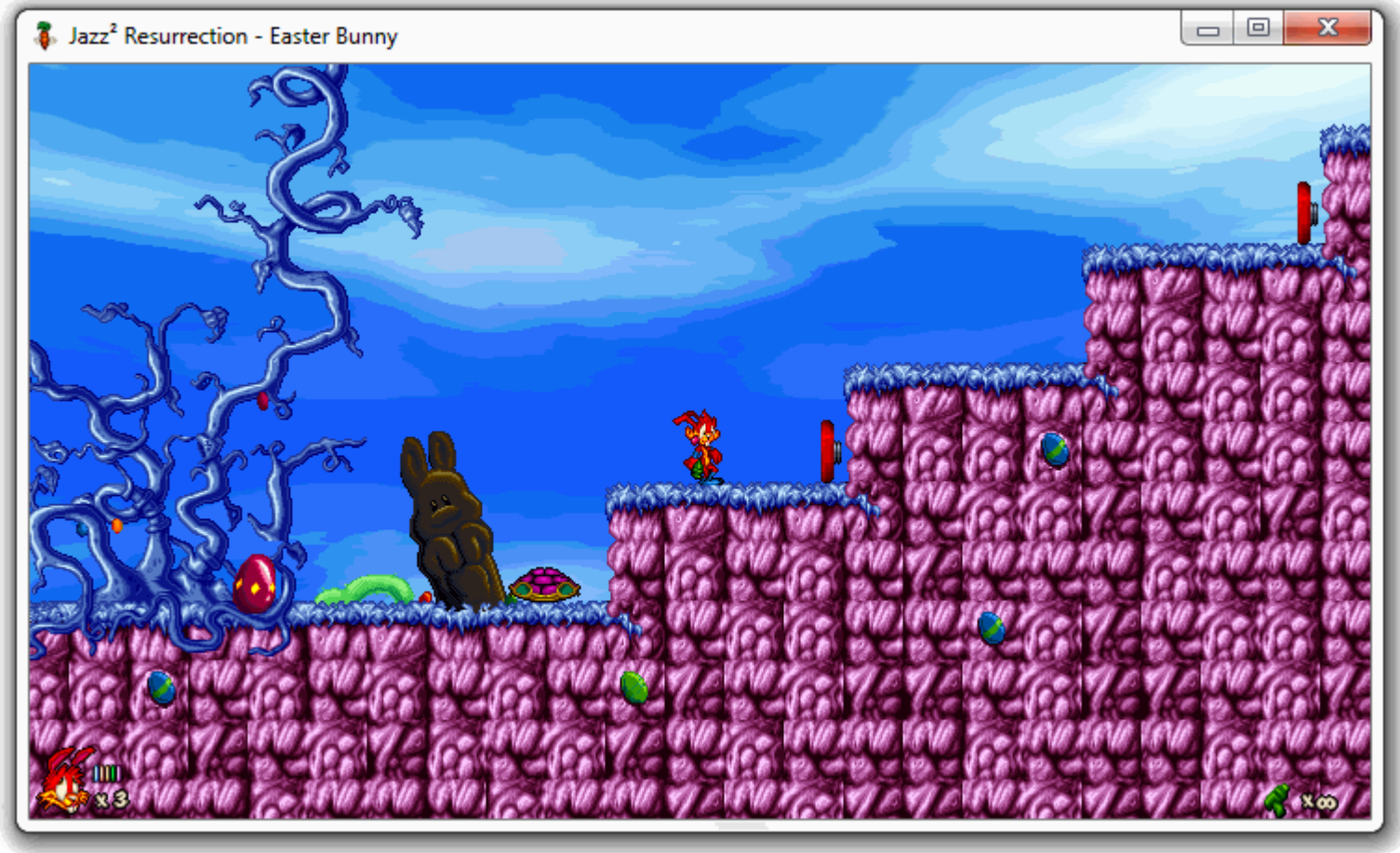
Android 5.0+ with OpenGL 3.0

Chrome, Edge, Firefox, ...

Experimental builds

Experimental builds (created directly from GitHub Actions) contains unreleased features, but may also contain work in progress or bugs. Please report bugs from these versions with exact version number.

Preview



Watch gameplay videos on **YouTube**

Running the application

Windows

- Download the game
- Copy contents of original *Jazz Jackrabbit 2* directory to `<Game>\Source\`
- Run `<Game>\Jazz2.exe` , `<Game>\Jazz2_avx2.exe` or `<Game>\Jazz2_sdl2.exe` application

`<Game>` is path to *Jazz² Resurrection*. The game requires **Windows 7** (or newer) and GPU with **OpenGL 3.0** support. Cache is recreated during intro cinematics on the first startup, so it can't be skipped.

Linux

- Download the game
- Install dependencies: `sudo apt install libglew2.2 libglfw3 libsdl2-2.0-0 libopenal1 libvorbisfile3 libopenmpt0`
 - Alternatively, install provided `.deb` or `.rpm` package and dependencies should be installed automatically
- Copy contents of original *Jazz Jackrabbit 2* directory to `<Game>/Source/`
 - If packages are used, the files must be copied to `~/.local/share/Jazz² Resurrection/Source/` instead
- Run `<Game>/jazz2` or `<Game>/jazz2_sdl2` application
 - If packages are used, the game should be visible in application list

`<Game>` is path to *Jazz² Resurrection*. The game requires GPU with **OpenGL 3.0** or **OpenGL ES 3.0** (ARM) support. Cache is recreated during intro cinematics on the first startup, so it can't be skipped.

Alternatively, you can use package repository for your Linux distribution:

Arch Linux

Flathub **v2.9.0**

Gentoo

NixOS

OpenSUSE passing

Ubuntu

macOS

- Download the game and install provided `.dmg` application bundle
- Copy contents of original *Jazz Jackrabbit 2* directory to `~/Library/Application Support/Jazz² Resurrection/Source/`
- Run the newly installed application

Cache is recreated during intro cinematics on the first startup, so it can't be skipped.

Alternatively, you can install it using  Homebrew `v2.9.1` `brew install --cask jazz2-resurrection`

Android

- Download the game
- Install `Jazz2.apk` or `Jazz2_x64.apk` on the device
- Copy contents of original *Jazz Jackrabbit 2* directory to `<Storage>/Android/data/jazz2.resurrection/files/Source/`
 - On **Android 11** or newer, you can *Allow access to external storage* in main menu, then you can use these additional paths:
 - `<Storage>/Games/Jazz² Resurrection/Source/`
 - `<Storage>/Download/Jazz² Resurrection/Source/`
- Run the newly installed application

`<Storage>` is usually internal storage on your device. `Content` directory is included directly in APK file, no action is needed. The game requires **Android 5.0** (or newer) and GPU with **OpenGL ES 3.0** support. Cache is recreated during intro cinematics on the first startup.

Nintendo Switch

- Download the game
- Install `Jazz2.nro` package (custom firmware is needed)
- Copy contents of original *Jazz Jackrabbit 2* directory to `/Games/Jazz2/Source/` on SD card
- Run the newly installed application with enabled full RAM access

Cache is recreated during intro cinematics on the first startup, so it can't be skipped. It may take more time, so white screen could be shown longer than expected.

Web (Emscripten)

- Go to <http://deat.tk/jazz2/wasm/>
- Import episodes from original *Jazz Jackrabbit 2* directory in main menu to unlock additional content




The game requires browser with **WebAssembly** and **WebGL 2.0** support – usually any modern web browser.

Xbox (Universal Windows Platform)


- Download the game
- Install `Jazz2.cer` certificate if needed (the application is self-signed)
- Install `Jazz2.msixbundle` package
- Run the newly installed application
- Copy contents of original *Jazz Jackrabbit 2* directory to destination shown in the main menu
 - Alternatively, copy the files to `\Games\Jazz² Resurrection\Source\` on an external drive to preserve settings across installations, the application must be set to `Game` type, `exFAT` is recommended or correct read/write permissions must be assigned
- Run the application again

Building the application


Windows

- Build dependencies will be downloaded automatically by *CMake*
 - Can be disabled with `NCINE_DOWNLOAD_DEPENDENCIES` option, then download  [Build dependencies](#) manually to `.\Libs\`
- Build the project with *CMake*
 - Alternatively, download  [Build dependencies](#) to `.\Libs\`, open the solution in  [Microsoft Visual Studio 2019](#) (or newer) and build it


Linux

- Build dependencies will be downloaded automatically by *CMake*
 - Can be disabled with `NCINE_DOWNLOAD_DEPENDENCIES` option, then download  [Build dependencies](#) manually to `./Libs/`
 - System libraries always have higher priority, there is no need to download them separately if your system already contains all dependencies
 - In case of build errors, install following packages (or equivalent for your distribution):
`libgl1-mesa-dev libglew-dev libglfw3-dev libsdl2-dev libopenal-dev libopenmpt-dev zlib1g-dev`
- Build the project with *CMake*

macOS

- Build dependencies will be downloaded automatically by *CMake*
 - Can be disabled with `NCINE_DOWNLOAD_DEPENDENCIES` option, then download  [Build dependencies](#) manually to `./Libs/`
- Build the project with *CMake*

Android

- Install Android SDK (preferably to `../android-sdk/`)
- Install Android NDK (preferably to `../android-ndk/`)
- Install Gradle (preferably to `../gradle/`)
- Build dependencies will be downloaded automatically by *CMake*
 - Can be disabled with `NCINE_DOWNLOAD_DEPENDENCIES` option, then download  [Build dependencies](#) manually to `./Libs/`
- Build the project with *CMake* and `NCINE_BUILD_ANDROID` option

Nintendo Switch

- Install [devkitPro toolchain](#)
- Build the project with *CMake* and devkitPro toolchain

```
cmake -D CMAKE_TOOLCHAIN_FILE=${DEVKITPRO}/cmake/Switch.cmake -D NCINE_PREFERRED_BACKEND=SDL2
```



Web (Emscripten)

- Install [Emscripten SDK](#) (preferably to `../emsdk/`)

```
cd ..
git clone https://github.com/emscripten-core/emsdk.git
cd emsdk
./emsdk install latest
./emsdk activate latest
```

- Build dependencies will be downloaded automatically by *CMake*
 - Can be disabled with `NCINE_DOWNLOAD_DEPENDENCIES` option
- Copy required game files to `./Content/` directory – the files must be provided in advance
- Build the project with *CMake* and Emscripten toolchain

Xbox (Universal Windows Platform)

- Build dependencies will be downloaded automatically by *CMake*
 - Can be disabled with `NCINE_DOWNLOAD_DEPENDENCIES` option, then download  [Build dependencies](#) manually to `.\Libs\`
- Run *CMake* to create  [Microsoft Visual Studio 2019](#) (or newer) solution

```
cmake -D CMAKE_SYSTEM_NAME=WindowsStore -D CMAKE_SYSTEM_VERSION="10.0"
```

License

This project is licensed under the terms of the [GNU General Public License v3.0](#) and uses modified [nCine](#) game engine.