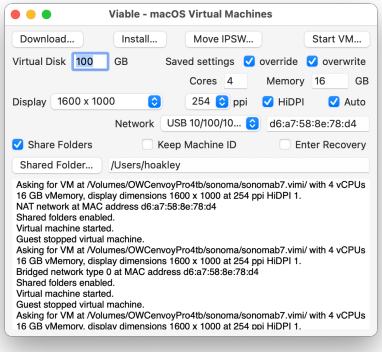
Start



Viable uses lightweight virtualisation in macOS Monterey 12.4 or later to build, install and run macOS virtual machines (VMs) on Apple silicon Macs. It can run one or two VMs simultaneously at speeds at or close to those of macOS running native. Once fully installed, its VMs can also be run using Vimy and enhanced using ARD.

Before building your first VM, you should:

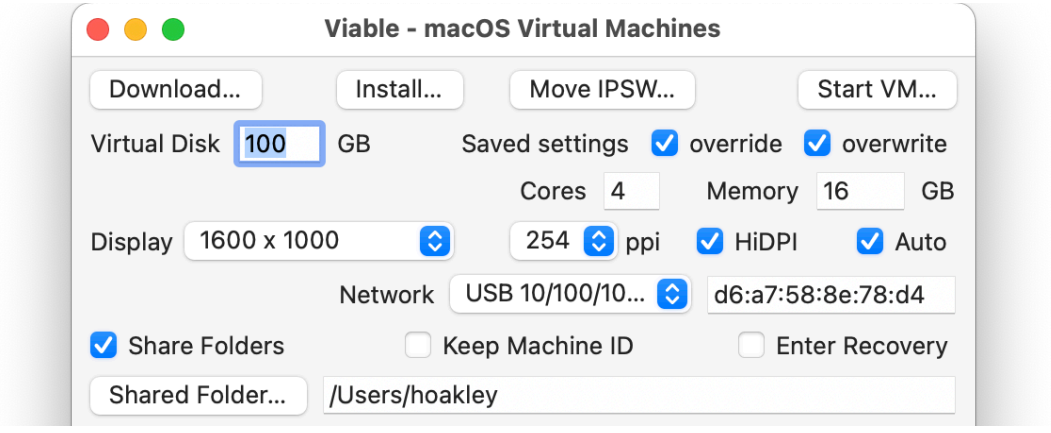
- *either* create a new folder, perhaps named VMs, and add that folder to your Mac's backup exclusion list,
- *or* create a new volume and add that to your Mac's backup exclusion list (preferred).

Storing your VMs in a folder saves them from being backed up every time they change. However, they'll still be included in any snapshots made of that volume. The second option ensures that they don't take any space in snapshots either.

To quit Viable, close its main window or use the **Quit** command.

➜ Controls ➜ Build a VM ➜ Run a VM ➜ Main settings ➜ Network settings ➜ Display settings
➜ Using shared folders ➜ VM features supported ➜ Enhanced VMs ➜ Machine IDs
➜ Problems ➜ Settings files ➜ Change list

Controls

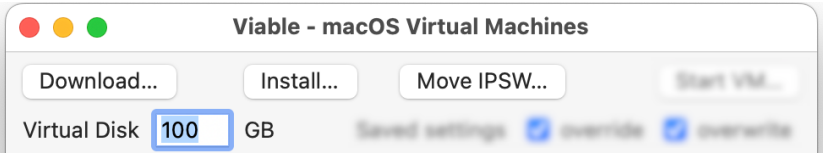


Viable's main window provides four buttons to perform its functions, each of which is also available from its **File** menu:

- **Download**: this downloads the latest release version of macOS as an IPSW image file, and uses that to install as a VM of the size in the **Virtual Disk** setting. Once it has completed that, the fresh VM is ready to personalise and configure on its first run using the **Start VM** button, and the downloaded IPSW image is saved inside the VM bundle.
- **Install**: this takes an IPSW image you have already downloaded from Apple and builds it into a fresh VM ready to personalise and configure on its first run using the **Start VM** button. This builds a VM of the size given in the **Virtual Disk** setting, with the IPSW image moved inside its bundle for safe keeping.
- **Move IPSW**: this extracts an IPSW file from inside a VM bundle so that you can easily save it somewhere else, perhaps to reuse in another VM. You can do this before starting the VM for the first time.
- **Start VM**: this runs a VM that you have already installed, using the settings in this window, or those already saved in the VM bundle, depending on the **Saved settings** you choose. The first time you start a VM that has been freshly installed, the macOS installer will take you through its initial configuration, including personalisation, just as you would when first setting up a new Mac.

➜ Build a VM ➜ Run a VM ➜ Main settings ➜ Network settings ➜ Display settings ➜ Settings files

Build a VM



You have two options to build a VM:

- to build one based on the current release of macOS, click the **Download** button. Viable will then try to download the IPSW image for the current release of macOS from Apple's servers. When that's complete, it will automatically be installed to build a fresh VM using that and the **Virtual Disk** setting. You don't need to use the **Install…** button in this case.
- to build one based on that or any another release of macOS Monterey or later, ensure you have the appropriate IPSW image file available, and click the **Install** button. You'll then be prompted to select the IPSW image file, and Viable will use that and the **Virtual Disk** setting to build and install a fresh VM ready for its first run.

The most comprehensive collection of links to Apple's IPSW images files is that of Mr. Macintosh.

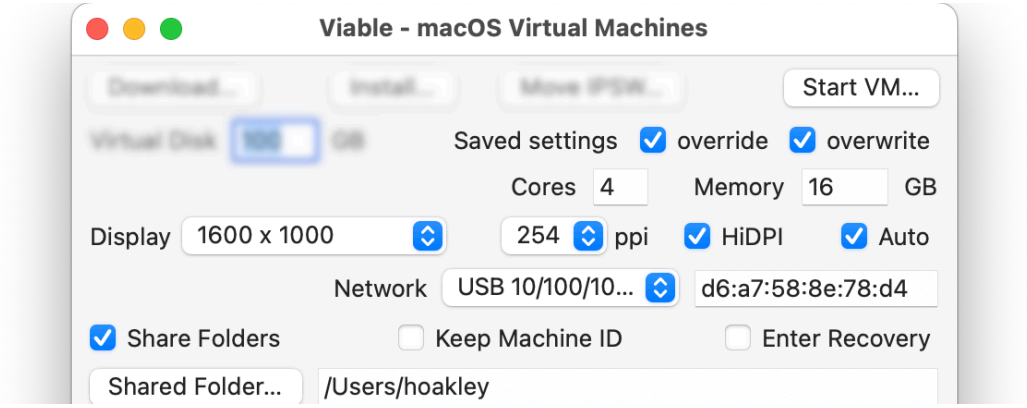
Settings used here are:

- **Virtual Disk**: this sets the size of the virtual block storage for the boot disk. Given in GiB, it can range between 15-2024, with a recommended minimum of 20 GiB. If you want to be able to update macOS within the VM, then a minimum size of 40 GiB is recommended, to ensure sufficient free space for updating. This size is fixed and can't changed after the VM has been installed.
- **Keep Machine ID**: this sets the machine identity of the VM to be the same as an ID already loaded. Normally, this should be left unchecked so that each VM has a unique ID. ➜ how to create VMs with the same machine ID.
- **Cores**, **Memory**, etc.: although these are used during the build process, they aren't set in the VM so can be whatever you like, within reason.

Save the VM using the .vimi extension so that it can be opened easily using Vimy.

Once a VM has been installed, and before or after it has been run the first time, you can extract its IPSW file and move it elsewhere using the **Move IPSW** button. You'll be prompted to select the VM, then choose a name and location for the IPSW file.

Run a VM



Once you have built, installed and saved a VM, the next task is to run it for the first time, so it can undergo initial personalisation and configuration, just as it would when running a fresh install of macOS. Viable can open bundles with the extension of .vimi (preferred) or .bundle. The first run of a VM is essentially the same as the first run of a new Apple silicon Mac, and requires similar personalisation.

⚠ Note that migration and iCloud aren't currently supported, and you should dismiss both of those during setup.

Before starting the VM, check how it's going to obtain its settings, and how it will save them, in the **Saved settings** checkboxes:

- **override**: when this is ticked, settings in this window will override any that may have been saved in the VM bundle. Use this the first time that you run a VM unless you want it to have default settings, and at any time you want to run a VM using new settings.
- **overwrite**: when this is ticked, the settings used (whether overridden or not) will be written inside the VM bundle, to be used the next time that VM is run with override not ticked. ➜ saved settings files.

To close a VM, 'shut down' the VM in the normal way, leaving the VM window black. The window will then close itself. The red Close button at the top left of the VM window *won't* close a running VM: to close the VM window, shut the VM down.

➜ Controls ➜ Build a VM ➜ Main settings ➜ Network settings ➜ Display settings ➜ Enhanced VMs

Main settings

Standard options supported are:

Cores: this specifies the number of cores on which the VM will be run, which is also the vCPU number. It can range between 1-20, with a recommended minimum of 2, and normally 4. The maximum used is limited to the number of cores in the host to prevent the cores from locking.

Memory: this specifies the size of memory to be given to the VM in GiB, and can range between 4-64, with a normal recommendation for 6-16.

Keyboard: the VM is configured to pass through special key features, such as the Globe key by default. When running on Sonoma hosts, additional keyboard support is also provided by default. No separate control is provided in this window.

When running any VM in Ventura and later, additional options are:
Enter Recovery: tick this box if you want the VM to start up in Recovery Mode.
Share Folders: tick this box if you want the VM to share folders with the host macOS.
Shared Folder: click on this button to select a folder to share between the host macOS and the guest.

➜ Run a VM ➜ Network settings ➜ Display settings ➜ Using shared folders ➜ Enhanced VMs

Network Settings

Viable supports two different types of network connection: NAT and bridged. While NAT is simpler, it's also not as fast, and is relatively inflexible. When using a NAT connection, your VM will have a local IP address assigned to it on a different sub-net, and packets will be transferred through the host Mac's network connection using address translation. Bridged networking shares one of the host's network ports, and gets its own IP address on your local network, as if it was another Mac.

Network popup menu: this offers a choice between bridged ports such as Ethernet and Wi-Fi, NAT, and no network connection. Where possible, you should prefer Ethernet; Wi-Fi may need additional configuration or may not work at all and fall back to Ethernet.

Address: set the desired NAT MAC address in this text box using hex notation. This defaults to Apple's standard of d6:a7:58:8e:78:d4. If you change it and want to return to the default, simply remove all text from the box and Viable will revert to the default for you. This is most important if you intend running two VMs at the same time. As they can't have the same MAC address, you should then alter the MAC address for one of them to a different setting. Otherwise it's best to leave this alone unless you understand what you're doing.

➜ Run a VM ➜ Main settings ➜ Display settings

Display settings

Display: this sets the native resolution of the virtual display, chosen from 2560 x 1440, 2200 x 1375, 1920 x 1200, 1728 x 1117, 1600 x 1000, 1512 x 982, and 1400 x 875 . Following those are pixel densities ranging from 72 to 256 pixels/inch, and a checkbox to determine whether to enable HiDPI.

HiDPI doubles the native resolution set, for example from 2560 x 1440 to 5120 x 2880. When your VM first boots at such a high resolution, windows may appear small. To obtain the best image possible on Retina displays, open the **Displays** pane/settings in the VM, select a Scaled resolution there labelled as **HiDPI** or the lower of the two offered, selecting **Show all resolutions** if necessary. Once set, so long as you open that VM with the same display settings, it will remain in that HiDPI resolution and deliver the best quality. Settings available in VMs depend on the version of macOS being run, and in some may not offer HiDPI at all.

There are two circumstances in which you may not want to run your VM in HiDPI: when running it for the first time after building it, and when booting in Recovery. In both cases that's because you can't then select HiDPI mode in the Displays pane, so everything will remain very small in your VM. Leaving the HiDPI checkbox unticked and using a lower resolution should give the best virtual display in those cases.

Currently, pixel densities don't appear to make any difference, and I suggest leaving that set at 80 or 254 ppi. When running a VM on a non-Retina display, you may need to experiment with different settings until you achieve the best image.

When running a Sonoma guest on a Sonoma host, the best option for display settings is **Auto** (not available on Monterey or Ventura), as that automatically rescales the display to the window size.

To adjust window size to suit virtual display resolution, use the **Snap** command in Viable's Window menu (not in the VM).

➜ Run a VM ➜ Main settings ➜ Network settings ➜ Using shared folders

Using shared folders

Shared folders are only supported when running Ventura or later as both guest and host. Four shares are provided as standard:

- Host Documents: ~/Documents
- Host Downloads: ~/Downloads
- Host Home: ~/
- VM Bundle: the top level inside the running VM bundle

The fifth, Custom, is set by the user using the **Shared Folder**… button.

Viable isn't sandboxed and its shared folders can be almost any folder on the host Mac. As even the default shared folders may require the user's consent before Viable can obtain access to the Documents and Desktop folders, you might find it useful to give Viable **full disk access** in **Privacy & Security** settings. If you don't, you'll prompted by macOS when it needs your consent to access a protected shared folder.

Access is provided to the VM bundle as that can make a VM entirely self-sufficient. If you create a new folder inside the VM bundle and use that for shared files, those files become part of the bundle, and will be copied and moved wherever that bundle goes. For example, you can use that to store working documents to which you'll retain access in each copy of that VM bundle.

While you can run apps from a shared folder, and can set /Applications as your VM's custom shared folder, it's generally best to copy all wanted apps to the guest /Applications folder before launching them from there. Sadly, you can't use this as a workaround to run third-party App Store or other apps that rely on your Apple ID.

ViableS is a sandboxed version of Viable without access to shared folders, intended for security research purposes.

VM features currently supported

Two concurrent VMs: given sufficient host resources, you can run two VMs at once. *Don't* try duplicating an existing VM to use for the second, as that will use an identical machine ID. Apple's licence limits the number to two, which is enforced by macOS.

Rosetta 2: Intel apps can be run within a VM. Additional software required to support Rosetta 2 is downloaded when required.

Bridged and **NAT network** connections. These work sufficiently to connect to the host using File Sharing (using smb://), to download and install macOS updates and security data updates, and for apps that need to access remote servers. It doesn't support iCloud connections at all, though.

Virtio and other devices: accelerated graphics, bridged and NAT networking, pointing devices (mouse, trackpad, with full support over Universal Control), Mac keyboard (ANSI layouts, or ISO with a missing key), audio (input and output), entropy (random number generation). Spice (clipboard) isn't yet supported as it appears incomplete. Although explicit trackpad support is used in Ventura and later, the Trackpad settings item isn't available in the VM.

Configurable CPU core count, memory, display resolution, network address.
Block storage.
Shared folders (Ventura and later only).
Creation of multiple VMs with the same machine ID.
Guest operating systems supported are Monterey, Ventura and Sonoma.
The ability to save and restore VM state in Sonoma isn't yet supported, but is coming in the future.

Note that the Virtualization framework doesn't currently support nesting VMs, in that you can't run Viable within a VM.

Enhanced features using ARD or VNC

Connect to a VM using Apple Remote Desktop (ARD) or the Screen Sharing app in /System/Library/CoreServices/Applications, started by entering the VM's IP address in the host's browser in a URL of the form `vnc://192.168.64.15`.

Ensure the VM has a network connection, and in its System Preferences or System Settings, open (General) Sharing. Tick the **Remote Management** item there, and tick all the boxes in its dialog for complete control as a VNC client for ARD or Screen Sharing.

In the host select the VM in ARD's **Scanner** view, and click the Control tool. After authenticating to it, use the ARD window containing the VM display and minimise the original VM window into the Dock.

You can now view and control the host in its ARD window, including:

- Drag and drop files between Finder windows in the VM and those on the host, to copy them.
- Change the keyboard on the **host** *(not the VM)* and use that layout; if the keyboard is in ISO format, the additional key at the upper left (under the Escape key) now works as you'd expect.
- Ensure the Clipboard tool in the ARD window is set to use the shared clipboard, and you can copy and paste between the ARD window on the VM and the host.
- Scale the ARD window on the VM to make that window smaller. You can't, though, use it to scale the contents of the window to a larger size.
- Use standard key shortcuts to make screenshots of windows or the display in the ARD window.
- Trackpad controls include gestures set on the host, and work smoothly and reliably with all guests.

These work uniformly across Monterey, Ventura and Sonoma VMs, even though Monterey VMs don't themselves support some of those features.

Shut the VM down using the **Shut Down** menu command in the ARD window on the VM, rather than in the original VM window.

Don't use ARD when you are going to update macOS in the VM, as that will disconnect ARD and could lead to problems.

➜ Run a VM ➜ Main settings

How to create VMs with the same machine ID

Normally, every VM created is assigned a unique machine identity. That ensures that, if you load and run two at once, they can't have the same ID, which would lead to "undefined behaviour". In other words, don't even try it. Unfortunately, virtualisation doesn't let you assign VMs specific serial numbers or UUIDs, but this is the closest you can come to that, as VMs with identical machine IDs have the same serial numbers and UUIDs.

Viable lets you create as many VMs as you wish with identical machine IDs, but it doesn't check whether you try to load two with the same ID: you have to be meticulously careful to avoid doing that.

Each time Viable starts up, it creates a unique machine ID you can use for this purpose. You can also load into memory any existing machine ID from another VM. With either of those, ticking the **Keep Machine ID** checkbox will keep using the same machine ID for all VMs that you create, until you untick the checkbox or quit the app.

To load a machine ID from an existing VM, use the **Load Machine ID…** command in Viable's **File** menu. In the file selector, load the file named *MachineIdentifier* from inside the VM bundle. If you want to build a library of such IDs for future use, simply copy those MachineIdentifer files into a folder, giving each an appropriate name. When Viable tries to load that file, if it isn't a proper machine ID you'll be informed, and you can try again.

You can only set machine IDs at the time you create a VM. Once it has been saved in the VM's **MachineIdentifier** file, it's fixed for that VM. Trying to change it by replacing the MachineIdentifier file inside the bundle will only cause that VM to freeze during loading, and it will no longer run until you replace its original MachineIdentifier file.

➜ Controls ➜ Build a VM ➜ Run a VM

When a VM goes wrong

The most common problem encountered with failing VMs normally occurs when they run out of resources, memory in particular. Rather than crashing, they usually grind to a halt and freeze. When this happens, close the VM window if you can, quit Viable, and start again, this time giving the VM more memory.

Another issue that can occur is trackpad/mouse/keyboard input behaves bizarrely, or stops working properly. For instance, you can't enter any text in Terminal. A simple trick which can solve that is to select Viable's main window to bring it to the front, then click/tap back in the VM's virtual display. If that doesn't work, try shutting the VM down as normally as possible, close its window, quit Viable, and try again.

One potential problem to beware of is using VMs which have simply been duplicated. This certainly causes problems when they're both run at once, as their machine IDs will be identical. Duplication is a good way to keep a copy of a VM, but not a good way to make a new VM quickly: you're better building a fresh VM.

➜ Controls ➜ Build a VM ➜ Run a VM ➜ Main settings ➜ Network settings ➜ Display settings

Saved settings files

When you first run a VM in Viable, it creates a file inside the VM bundle containing the settings used to run that VM. Whenever you run that VM after that, you can either override those settings with those in Viable's window, or use the saved settings for that VM. Those are the settings used automatically when you run that VM using Vimy, where they can't be overridden or overwritten.

Current saved settings are stored in a property list file named `Settings.plist` inside the VM bundle. As that file *isn't* managed by `cfprefsd`, and is only read when running a VM, you are free to edit it manually using a suitable editor if you wish.

When you first overwrite saved settings in a VM, the previous settings are copied to a backup settings file named `PreviousSettings.plist`. Only one backup settings file is kept, and you can make that the current settings file simply by renaming it `Settings.plist` if you want.

Unlike machine IDs, settings files aren't unique to a VM, and you could copy them between VM bundles if you wish.

➜ Controls ➜ Build a VM ➜ Run a VM ➜ Main settings ➜ Network settings ➜ Display settings

Change list

1.0.12 (beta 12):

- fixed a bug that displayed the file open dialog when restoring a VM from a minimised window.

1.0.11 (beta 11):

- added bridged networking
- added network popup menu
- improved transcript in scrolling text view.

1.0.10 (beta 10):

- corrected Help book
- added advice to Help book on use of ARD
- fix bug that prevented custom MAC addresses from being used
- improved error messages.

1.0.9 (beta 9):

- added vimi document type
- updated UTType support
- added saved settings features
- redesigned main window, removing redundant keyboard items
- customised menus to buttons
- added support for moving IPSW files
- changed VM info on launch
- replaced RTF help with PDF
- added full tooltips.

1.0.8 (beta 8):

- added support for auto sizing of displays (Sonoma)
- added support for keystroke pass through
- added support for new Mac keyboard (Sonoma)
- added support for custom network addresses
- disable VM window closing while the VM is still running
- improved saving of VM settings.

1.0.7 (beta 7):

- refactored the VM to handle errors more gracefully
- automatically close VM windows when they shut down

- added Shared Folders checkbox (macOS 13 hosts)
- further improvements to UTI/bundle handling
- improved information about VM when it's run
- corrected text for HiDPI checkbox.

1.0.6 (beta 6):

- restored the default .bundle extension when saving VM bundles.

1.0.5 (beta 5):

- revised and fixed shared folder support for Ventura
- added full support for the trackpad in Ventura
- removed the sandbox
- prepared clipboard support, although it remains non-functional.

1.0.4 (beta 4):

- added support for creating VMs with the same machine ID.

1.0.3 (beta 3):

- added more display resolutions
- added HiDPI option
- augmented text output.

1.0.2 (not released):

- added ppi popup menu
- tried (unsuccessfully) shared folder support in macOS 12 host.

1.0.1 (beta 2):

- added Snap command to Window menu, to resize windows
- added 1400 x 875 virtual display resolution
- rerouted errors to report to app main window.

1.0 (beta 1):

- first beta release.

22 November 2023.

➜ Additional licence

Additional licence

The following licence applies to the source code incorporated from Apple's sample code into the source code of Viable:

Copyright © 2022 Apple Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Acknowledgement

I’d like thank Tobias Bussmann who suggested using ARD to enable enhanced features.