Raspberry Pi Kiosk using Chromium

by Gus - https://pimylifeup.com/raspberry-pi-kiosk/
Updated 28. 10. 2022

For this Raspberry Pi Kiosk tutorial, we will be showing you how you can set up your Pi as a Kiosk using the popular web browser, Chromium.

- - - - - - - -

We utilize Chromium as it is one of the best-supported web browsers and openly supports the functionality to act in a kiosk mode.

It is also easy to control through key presses which we can simulate using the **xdotool** that we install during this guide.

This guide will give you a good idea on how to use autologin tasks and simple bash scripts to perform numerous tasks.

Please note to do this tutorial you will need to be running the full version of Raspbian as this relies on the GUI that comes with it to properly operate.

Equipment List

The entire list of all the pieces of equipment that we made use of for this Raspberry Pi Kiosk tutorial is listed below.

Recommended

- Raspberry Pi 2, 3, 4 or newer
- Micro SD Card
- Power Supply
- TV or Monitor
- HDMI Cable
- Ethernet Cable or Wi-Fi

Optional

- Raspberry Pi Case

Video Tutorial

In this video, we show you the process of setting up your Raspberry Pi to operate as a Chromium powered kiosk as well as showing you how to get it to start at boot.

If you prefer a written and more thorough explanation, then check out our written guide below.

Adblock removing the video? Support us by subscribing to our ad-free service.

Setting up the Raspberry Pi Kiosk

1. Before we get started with this tutorial, we will be first removing some packages that we don't need for our Raspberry Pi kiosk.

Removing these packages will free up some much-needed memory and reduce the number of packages that will be updated every time you update your Raspberry Pi.

To do this just run the following three commands on your Raspberry Pi. We have split these into three different commands to make them easier to copy and write out.

```
sudo apt purge wolfram-engine scratch scratch2 nuscratch sonic-pi idle3 -y
sudo apt purge smartsim java-common minecraft-pi libreoffice* -y
```

2. Now that we have removed these numerous packages from our Raspberry Pi Kiosk we will need to run some cleanup commands.

To remove any unnecessary lingering packages and to clean out the local repository of retrieved packages run the following two commands on your Raspberry Pi.

```
sudo apt clean
sudo apt autoremove -y
```

3. Now that we have removed the bulk of the unnecessary applications, we must now make sure that our installation of Raspbian is up to date. Also, make sure you have SSH enabled as it is very handy if you need to edit any files later on.

We can use the following two commands within the terminal on our Raspberry Pi to update all the packages to their latest versions.

```
sudo apt update
sudo apt upgrade
```

4. We now need also to install xdotool. This tool will allow our bash script to execute key presses without anyone being on the device. We also install the unclutter package, this will enable us to hide the mouse from the display.

Just run the following command on your Raspberry Pi to install the package.

```
sudo apt install xdotool unclutter sed
```

5. Now that those packages have been installed we can now proceed to the next stage of this tutorial. That is setting up **Raspberry Pi OS to auto login** to our user. Having to log in every time for a kiosk would be an annoyance.

Desktop autologin is the default behavior but if for some reason you have changed it follow the next few steps to switch it back. Otherwise, skip to **step 6** of this tutorial.

5a. Run the following command on your Raspberry Pi to load up the Raspi-config tool. We will be using this tool to enable auto login.

```
sudo raspi-config
```

5b. Now within the tool go to **1 System Options** -> **S5 Boot / Auto Login** -> **B4 Desktop Autologin**

5c. Desktop autologin should now be enabled and you can safely quit out of the **raspi-config** tool.

6. Now that we have enabled desktop autologin we need to go ahead and write our **kiosk.sh** script.

Writing the Raspberry Pi Kiosk Script

The kiosk script will handle the bulk of the work for our Raspberry Pi Kiosk, including launching Chromium itself and also simulating key presses.

1. Begin writing our kiosk bash script by running the following nano command on the Raspberry Pi.

```
nano ~/kiosk.sh
```

2. Within this file enter the following lines of code, we will explain the important parts of the script so you can better bend it to your needs, kiosk.sh – Line 1:

```
#!/bin/bash
```

The very first line defines what the command line interpreter (CLI) should use to try and execute this file.

This is useful for cases where you don't want to have to specify the specific application required every time you run the script, kiosk.sh - Line 2 – 4:

```
xset s noblank
xset s off
xset -dpms
```

These three lines are pretty important as they help us stop the Raspberry Pi's display power management system from kicking in and blanking out the screen.

Basically, these three commands set the current xsession not to blank out the screensaver and then disables the screensaver altogether.

The third line disables the entire "display power management system" meaning that the desktop interface should never blank out the screen, kiosk.sh - Line 5:

```
unclutter -idle 0.5 -root &
```

This line runs the program we installed earlier called **unclutter**.

This application will hide the mouse from the display whenever it has been idle for longer then **0.5** seconds and will remove it even if it is over the root background.

You can adjust the idle timer to the number of seconds you want with each decimal place being a fraction of a second.

If you would prefer to remove the mouse instantly, then remove the **`-idle 0.5`** from the command, kiosk.sh - Line 6 – 7:

```
sed -i 's/"exited_cleanly":false/"exited_cleanly":true/'
/home/$USER/.config/chromium/Default/Preferences
sed -i 's/"exit_type":"Crashed"/"exit_type":"Normal"/'
/home/$USER/.config/chromium/Default/Preferences
```

These two lines of the script use **sed** to search through the Chromium preferences file and clear out any flags that would make the warning bar appear, a behavior you don't really want happening on your Raspberry Pi Kiosk.

If Chromium ever crashes or is closed suddenly, the lines above will ensure you don't have to get hold of a mouse and keyboard to clear the warning bar that would typically appear at the top of the browser, kiosk.sh - Line 8:

```
/usr/bin/chromium-browser --noerrdialogs --disable-infobars --kiosk
https://pimylifeup.com https://www.adafruit.com &
```

This line launches Chromium with our parameters.

We will go through **each of these parameters** so you know what you can modify, and how you can modify it. Enable Chromium Kiosk Mode:

```
--kiosk
```

This flag sets Chromium to operate in Kiosk mode, this locks it into a particular set of features and only allows limited access to both the web browser and any other OS functionality.

Chromium's kiosk functionality takes full control of the screen, maximizing Chromium to the full size of your screen and stops user input from being accepted by the OS, effectively trapping the end user within a sandbox. Disable Error Dialog:

```
--noerrdialogs
```

This option tells Chromium that it should not display any of its error dialogs to the end user.

It is crucial if you don't want the end user to know if anything has or is going wrong with Chromium, this goes alongside our code to clear the "**exited_cleanly**" and "**exit_type**" state earlier in the code. Disable Chromium Info Bars From Appearing:

```
--disable-infobars
```

We use this to disable Chromium from displaying its info bar to the end user.

The info bar can be used by Chromium to notify them of certain things such as that Chromium is not their default web browser.

Of course, as we are using this as a kiosk, we don't need the user to know any information that Chromium might want to display. Pages to Load in Kiosk:

```
https://pimylifeup.com https://www.adafruit.com
```

These are the two web pages that the script will open, each will be opened in a new tab.

You can add additional web pages/tabs by adding to this list by separating each one with a space, kiosk.sh – Lines 9 – 12:

```
while true; do
        xdotool keydown ctrl+Next; xdotool keyup ctrl+Next;
    sleep 15
done
```

These lines run a very simple infinite while loop that uses **xdotool** to mimic pressing `CTRL` + `Pgdn`. Making Chromium switch to the next tab. "**`Next`**" is a alias for the "**`PG DN`**" key.

After **xdotool** has executed its key presses, it then puts the loop to sleep for 15 seconds.

To change how long the loop sleeps for before it executes the xdotool command again just change the **`sleep 15`** command.

You can also use this method to add a screen refresh, this may be important when you want to display live scores.

The command for this should look something like what we have shown below, kiosk.sh – Alternate Refresh Command:

```
xdotool keydown ctrl+r; xdotool keyup ctrl+r;
```

3. Once you have entered all the code for our Raspberry Pi kiosk script, it should look somewhat similar to what we have displayed below. Add:

```
#!/bin/bash
xset s noblank
xset s off
xset -dpms

unclutter -idle 0.5 -root &

sed -i 's/"exited_cleanly":false/"exited_cleanly":true/'
/home/$USER/.config/chromium/Default/Preferences
sed -i 's/"exit_type":"Crashed"/"exit_type":"Normal"/'
/home/$USER/.config/chromium/Default/Preferences

/usr/bin/chromium-browser --noerrdialogs --disable-infobars --kiosk
https://pimylifeup.com https://www.adafruit.com &

while true; do
   xdotool keydown ctrl+Next; xdotool keyup ctrl+Next;
   sleep 10
done
```

4. Once you are sure everything is correct, save the file by pressing `CTRL` + `X` then `Y` and finally `ENTER`.

5. After creating this script we should make sure that our user has execution privileges for it.

You can give the user that owns the script execution privileges by running the following command.

```
chmod u+x ~/kiosk.sh
```

Setting up the Raspberry Pi Kiosk to start at boot

1. Before we get started we need to first utilize the following command to work out what the current display value is.

This value is used for the operating system to know what screen to display the Chromium kiosk to, without it the kiosk will either fail to load or load up on the incorrect screen.

Run the following command to echo out the value of the "**`$DISPLAY`**" system variable.

```
echo $DISPLAY
```

Make sure you remember this value as you may need it in **step 3** of this section.

2. To get our Raspberry Pi Kiosk to start at boot we will need to go ahead and create a service file by running the command below.

This service file will tell the operating system what file we want to be executed as well as that we want the GUI to be available before starting up the software.

```
sudo nano /lib/systemd/system/kiosk.service
```

3. Within our kiosk service file, enter the following lines of text.

These lines are what will define our service kiosk service, and that we want to run our kiosk.sh script when the system loads into the operating system.

While entering these lines you may have to modify the "**`Environment=DISPLAY=:`**" line, replacing the "**`0`**" with the value that you retrieved from the command you used in **step 1** of this section.

Additionally, you will need to make sure you replace "**`pi`**" with the name of your user. For example, with the username "**`pimylifeup`**", the path "**`/home/pi/kiosk.sh`**" would become "**`/home/pimylifeup/kiosk.sh`**". Add:

```
[Unit]
Description=Chromium Kiosk
Wants=graphical.target
After=graphical.target

[Service]
Environment=DISPLAY=:0.0
Environment=XAUTHORITY=/home/pi/.Xauthority
Type=simple
ExecStart=/bin/bash /home/pi/kiosk.sh
Restart=on-abort
User=pi
Group=pi

[Install]
WantedBy=graphical.target
```

Once you have entered everything into the file, save the file by pressing **`CTRL`** + **`X`** followed by **`Y`** then **`ENTER`**.

4. Now that we have created the service file for our Raspberry Pi Kiosk we can go ahead and now enable it by running the following command.

By enabling the service, we will allow our Chromium Kiosk to start up at boot automatically and will enable the systemd to service manager to monitor it.

```
sudo systemctl enable kiosk.service
```

5. With the Kiosk service now enabled you can either choose to restart the Raspberry Pi or start the service now by running the following command.

```
sudo systemctl start kiosk.service
```

6. If you ever want to check the status of your Raspberry Pi Kiosk's service, you can run the command below.

This command will return various information about the service, including previously returned lines from the software which can help you debug what's going wrong when the service is failed.

```
sudo systemctl status kiosk.service
```

If this command shows the status as "**Active: active (running)**" then everything is now working as it should be, and your Raspberry Pi Chromium Kiosk should be up and operating correctly.

7. Now with everything up and running correctly, if there is for any reason you want to stop the service from running, you can utilize the following command.

```
sudo systemctl stop kiosk.service
```

By stopping the kiosk service, the service manager will kill off all processes associated with it.

This command will stop our **`kiosk.sh`** script from running while also terminating the open Chromium browser.

8. Finally, if you ever want to disable your Kiosk, you can utilize the following command.

```
sudo systemctl disable kiosk.service
```

This command will stop the Kiosk service from running on boot until you re-enable it.

Enforcing the Resolution on a Raspberry Pi Kiosk

1. One thing you might want to do is to enforce the resolution that the Raspberry Pi is going to use. Setting the resolution can be quite handy as the Raspberry Pi's built-in detection can sometimes be a bit flakey.

To begin setting the resolution, we must first load up the Raspberry Pi configuration tool by running the following command.

```
sudo raspi-config
```

2. Within the configuration tool, you will want to start by going to "**7 Advanced Options**".

3. Now that we are in the advanced options section you should see an option labeled "**A5 Resolution**" select that option.

4. Within here find and select the resolution that best fits your screen and press **`ENTER`**.

5. With your resolution now set you will need to restart the Raspberry Pi. Do this by first exiting out of the configuration tool by pressing **ESC**, then entering the following command into the Raspberry Pi's terminal.

```
sudo reboot
```

6. Your Raspberry Pi should now restart and be running at the specified resolution.

Conclusion

There are so many ways you can extend this tutorial. For example, you can set up a web server on the Raspberry Pi and have it serve local web pages to be displayed on your Kiosk.

It's perfect if you want the results of a competition or basically any other sort of information you want to be displayed.

By now you should have your Raspberry Pi successfully booting into the Kiosk mode of Chromium. If you have any issues with this Raspberry Pi kiosk tutorial or want to leave feedback, then feel free to leave a comment below.

Weekly Updates Straight To Your Inbox

Receive our Raspberry Pi projects, coding tutorials, Linux guides and more!

121 Comments

1. **Brent** on

   Loving your tutorials. Would this same functionality also work on a Pi Zero W?

1. **Emmet** on

1. Hi Brent,

   This tutorial should work on a Pi Zero W. Just try not to open too many tabs within the kiosk as Chromium can be fairly heavy.

   Cheers,
   Emmet

- **Benny** on

Hi Emmet,
Thanks so much for the great description. I'm running this solution on an RPI4. The strange thing was that sometimes after a reboot chromium was not started and the status showed that there was a problem opening the display. Because the issue was 'flaky', i thought it might be a timing issue. After adding a delay in the service the problem was resolved. I used 20 seconds but i think that is much too long. But for me it's fine.
The service now is as follows:

```
[Unit]
Description=Chromium Kiosk
Wants=graphical.target
After=graphical.target

[Service]
ExecStartPre=/bin/sleep 20
Environment=DISPLAY=:0.0
Environment=XAUTHORITY=/home/dorpshoes/.Xauthority
Type=simple
```

```
ExecStart=/bin/bash /home/dorpshoes/kiosk.sh
Restart=on-abort
User=dorpshoes
Group=dorpshoes

[Install]
WantedBy=graphical.target
```

- **Anne-Laure** on

Thanks for this great tutorial. I was able to display a web site on a screen using a headless Raspberry Pi 3 with a Rasbian Lite OS.

- **Brittany Villaseñor** on

Thanks for the awesome tutorial. I am trying to get the data on my page to automatically refresh to show more of that “live” data you mentioned. I tried to use the following:

```
while true; do
xdotool keydown ctrl+r; xdotool keyup ctrl+r
sleep 10
```

but I used it in place of the ctrl+Next. Am I doing this wrong? Was I supposed to add that to the script and not replace it? It ended up getting my webpages stuck on the first page when I used the ctrl+r. I even tried ctrl+f5 and that did the same thing. I’m an intern trying to make a good impression, so any help is appreciated!

1. **Emmet** on

1. Hi Brittany,

   Sorry to hear you are having troubles with the tutorial and hopefully I’ll be able to help out.

   If I’m not mistaken, you are attempting to change to the next tab, while also then performing a refresh so that the data on the screen is updated?

   If that is the case, then you will want to retain the original xdotool calls, adding the alternative refresh one below it as we have shown below.

   ```
   while true; do
      xdotool keydown ctrl+Next; xdotool keyup ctrl+Next;
      xdotool keydown ctrl+r; xdotool keyup ctrl+r;
      sleep 10
   done
   ```

   Please let me know if this resolves the problems that you are running into, and best of luck with your internship.

   Cheers,
   Emmet

- **Peter** on

Hi Emmet
I’m hoping you can help me, I set up the kiosk.sh and it works perfectly.
I then set up the kiosk.service and it boots to the home screen. I ran the status on it and it says it has failed. ExecStart=/bin/bash /home/pi/kiosk.sh (code=exited, status=1/FAILURE) It later says xset: unable to open display””

1. **Emmet** on

1. HI Peter,

   Sorry for the late reply.

   Have you made sure that the “Environment=DISPLAY=:0.0” value within the service file matches the value you are getting from the “echo $DISPLAY” (Command needs to be ran on the Pi directly) command?

   Cheers,
   Emmet

- **Balazs** on

Great tutorial, Thank you!

- **Ash** on

Great tutorial! Thank you. I have the Raspberry Pi 1GB model and for some reason when chromium launches, it keeps freezing to a white screen on load. Is there a way to fix it? Thanks

1. **Emmet** on

1. Hi Ash,

   What particular Raspberry Pi model are you using? Are you sing the Pi 1, or a newer model like the Pi 4?

   Chromium can be fairly memory intensive so there is a chance that they little amount of memory is being gobbled up instantly, causing your issues.

   Cheers,
   Emmet

2. **Patrick** on

   I had a similar problem. Try to enable or disable the hardware acceleration in the settings of chromium. This fixed my problems when i faced a white screen when opening my page in chromium kiosk mode.

3. **Ash** on

Just realised my raspberry pi 3 A+ only has a memory of 0.5GB of memory. I have upgraded to the B+ with a memory of 1GB and turned off hardware acceleration on chromium. Now it's fully working, thanks!

- **Mark McKeel** on

Hi,
Great tutorial. I tried this back in 2019 and it worked flawlessly. Re-visited in 2022 and now can't get more than one page to load. Have checked the URLs and all load outside of the kiosk but when using the script only one page loads. Verified but hitting ctr+Tab from keyboard while in kiosk mode and nothing changes. Any thoughts?

Thanks

1. **Emmet** on

1. Hi Mark,

   It looks like that the latest versions of Chromium no longer uses CTRL + TAB to change tabs.

   We have updated the tutorial to use an alternative keystroke to change the tabs, CTRL + PgDN.

   Please let me know if this solves your issue.

   Cheers,
   Emmet

- **Kit Eason** on

Some things I found in 2022:

– Chrome no longer seems to use Ctrl+Tab to switch between tabs (!). So instead in your kiosk.sh:

```
while true; do
xdotool keydown ctrl+Next; xdotool keyup ctrl+Next;
sleep 15
done
```

This uses an alias of the PgDn key called Next.

– As others have hinted, you'll definitely have to turn on execution for your kiosk.sh:

```
chmod u+x kiosk.sh
```

– I found that sometimes the web pages would get shown before the network was up, leading to visible errors. I added a

```
sleep 10
```

above the while loop in kiosk.sh.

-

Will on

Hi,

I have set up an apache server on our LAN network that is accessible on the chromium browser of my Raspberry PI if I open it manually, but if it is opened through the kiosk.service it just results in a blank screen.
I have to accept that the site uses an invalid certificate when I go to the site manually, could this be the source of the problem?

1. **Emmet** on

1. Hi Will,

   Sorry for the late reply.

   It is likely that the certificate errors is causing part of your issue. You can work around this by including two additional command options.

   Those being the following. Please use these with caution though as it will causing Chromium to ignore all certificate errors.

   `--ignore-certificate-errors --ignore-urlfetcher-cert-request`

   Please let me know if adding those two options help with the blank screen problem you are running into.

   Cheers,
   Emmet

- **Leonardo** on

Thanks for the guide, I have a problem..

my url has an & in it and chromium only opens before the & part

ex: `www. site&Lang=esp&Type=comedy`
and only opens
`www. site`

1. **Emmet** on

1. Hi Leonardo,

   Please try wrapping the URL in double quotes (") like so, `"https://pimylifeup.com/?test&hello"`.

   Alternatively, you can also escape the ampersand (&) by using a backwards slash (\) before the symbol. For example, `https://pimylifeup.com/?test\&hello`.

   Cheers,
   Emmet

- **Rudy Mekkes** on

It works great but is there a Key-combination to exit/stop the kiosk?

1. **Emmet** on

1. Hi Rudy,

   With the current setup there is no key combination. You would have to SSH into your device and stop the service to stop it from reopening the web browser.

   Cheers,
   Emmet

2. **Rudy Mekkes** on

   Hi Emmet, thanks for your quick response.
   (I think) I solved the problem bij activating both HDMI screens.
   Your Kiosk is running on HDMI-1 and the desktop is running on HDMI-2
   VNC shows both screens, so I am able to work thru the GUI.

- **Bonzadog** on

An interesting project but I prefer using another browser han Chromium….would this be possible?

1. **Emmet** on

1. Hi Bonzadog,

   The biggest problem I've ran into personally is that most other browsers for the Raspberry Pi don't seem to offer kiosk functionality built in.

   Firefox is meant to but I think the ESR releases that are available for the Pi are to old to have that functionality.

   Cheers,
   Emmet

- **Skoobi** on

Hi. Great tutorial, helped me a lot and all worked perfectly. I am looking for a way to disable tabs or allowing the ctrl+t to open a new tab, is there a way to disable certain features?

Cheers

- **Price** on

This has worked really well for me. Thank you! Is there a way to do this in Incogneto?

1.

Emmet on

1. Hi Price,

   Yes there is definitely a way. All you need to do is add `--incognito` to the the Chromium browser call.

   So for example the following.

   ```
   /usr/bin/chromium-browser --noerrdialogs --disable-infobars --kiosk https://pimylifeup.com https://www.adafruit.com &
   ```

   Would become

   ```
   /usr/bin/chromium-browser --noerrdialogs --disable-infobars --incognito --kiosk https://pimylifeup.com https://www.adafruit.com &
   ```

   Cheers,
   Emmet

- **Local-E** on

I am using this for a menu board for a hosted html page. When I run the kiosk mode the font formatting seems to change and become bold, and then no longer space correctly. When I load Chromium when after disabling the kiosk mode, the page will load properly. I also had to use the Style=forking instead of Style=simple to get the kiosk mode to work.

Any thoughts on what is messing with the formatting when kiosk mode is activated?

1. **Emmet** on

1. Hi Local-E,

   To get a better idea of what could possibly be going wrong we would need to see a sample of the HTML / CSS that you are using.

   It could possibly be that for some reason Chromium is sizing differently when its put in to kiosk mode.

   Cheers,
   Emmet

- **Jeff** on

One thing I noticed, that may help others: if you're getting the 'cannot open display' problem or similar issues, check the boot options in raspi-config. If it's graphical desktop with login prompt, then change it to graphical desktop with autologin.
The X display doesn't belong to the 'pi' user until that user logs in, so the service running as 'pi' will be refused access to the display if it's showing the login page.
An alternative, if you don't want autologin, might be to run the service as root, which should own the display pre-login; I've not tested that and I'd avoid it because of the security risks.