Setup a Raspberry Pi to run a Web Browser in Kiosk Mode

This guide provides a very lightweight setup for a Raspberry Pi in kiosk mode: Instead of stripping down a full desktop environment like PIXEL or LXDE, we start without any GUI at all and install only the bare minimum needed to display a web browser in full screen.

Start With a Fresh Install of Raspian Lite

Download and install Raspbian **Lite**. In contrast to Raspian Desktop, Raspian Lite has no desktop environment preinstalled and is generally much lighter and smaller in size.

This guide is based on the November 2017 version of Raspian (*Raspbian Stretch Lite*), but it should work for other versions, too.

Boot up the Raspberry Pi, login as user `pi` with password `raspberry`, then start `sudo raspi-config` to apply some initial customizations:

- **Localisation Options:** Select your preferred locale (we simply keep the default `en_GB.UTF-8`), timezone, and keyboard layout.
- **Change User Password:** This is **important** – keeping the default password means your Pi will get owned faster than you can say “botnet” as soon as you connect it to the internet. (Make sure to have selected the correct keyboard layout before typing in the new password, though.)
- **Network Options:** Configure WiFi as needed. Alternatively, you also configure WiFi manually using `wpa_passphrase` if you don’t want your WiFi password stored on the Pi in clear text.
- **Boot Options:** Select “Desktop / CLI” and then “Console Autologin”. We’ll come back to this later.
- **Interfacing Options:** Enable SSH access if needed.
- **Advanced Options:** Disable “Overscan” if the Pi’s output does not fill your screen completely.

Now reboot the Pi. If everything was done correctly you should end up in a terminal session without having to enter your password.

To conclude the initial setup, update all preinstalled packages:

```
1 sudo apt-get update
2 sudo apt-get upgrade
```

Minimum Environment for GUI Applications

Usually the graphical environment for GNU/Linux consists of four parts:

1. X server (usually X.Org)
2. Window manager (Openbox, XFWM, …)

3. Desktop environment (PIXEL, LXDE, MATE, …)
4. Login manager (for example LightDM)

However, we only want to run a single application (the web browser) in full screen – so we don't need a desktop environment. And we already have autologin enabled (and no other users will ever use the Pi) – so we don't need a login manager either.

The bare minimum we need are X server and window manager. Let's install just that:

```
sudo apt-get install --no-install-recommends xserver-xorg x11-xserver-utils xinit openbo
```

Web Browser

We'll use Chromium because it provides a nice kiosk mode:

```
sudo apt-get install --no-install-recommends chromium-browser
```

Openbox Configuration

Now with everything in place, we can configure Openbox. Edit `/etc/xdg/openbox/autostart` and replace its content with the following:

```
# Disable any form of screen saver / screen blanking / power management
xset s off
xset s noblank
xset -dpms

# Allow quitting the X server with CTRL-ATL-Backspace
setxkbmap -option terminate:ctrl_alt_bksp

# Start Chromium in kiosk mode
sed -i 's/"exited_cleanly":false/"exited_cleanly":true/' ~/.config/chromium/'Local Stat
sed -i 's/"exited_cleanly":false/"exited_cleanly":true/; s/"exit_type":"[^"]\+"/"exit_t
chromium-browser --disable-infobars --kiosk 'http://your-url-here'
```

First we disable screen blanking and power management (we don't want our screen to go blank or even turn off completely after some time).

Then we allow to quit the X server by pressing `Ctrl-Alt-Backspace`. (Because we didn't install a desktop environment there won't be a "Log out" button or the like.)

Finally we tell Openbox to start Chromium in kiosk mode. This turns out to be a bit intricate because Chromium loves to show various tool bubbles for session restore etc. The simplest way to avoid all of these seems to be tricking Chromium into thinking it exited cleanly last time it was run (see this answer on Super User for details).

That's it! Time to give it a try:

```
startx -- -nocursor
```

After a few seconds Chromium should appear showing the URL you specified. Oh, and as you might have guessed: The `-nocursor` option tells X to not display any mouse cursor at all.

Press `Ctrl-Alt-Backspace` to quite the X server, bringing you back into the text console.

Start X automatically on boot

Now there's only one thing left: The X server should start automatically on boot.

Because we already configured the Pi to autologin the `pi` user, we can use its `.bash_profile` for starting X. Simply append the following line:

```
1 [[ -z $DISPLAY && $XDG_VTNR -eq 1 ]] && startx -- -nocursor
```

The condition makes sure that X is only started on the first console (and if it isn't already running). Because autologin uses the first console, this has the desired effect of automatically starting the X server (and thus the window manager and thus Chromium) on boot. And you can still use any of the other consoles for logging in manually.

Reboot your pi to test if everything works as expected.

Usage tips

- If Chromium (or the X server) crashes, press `Ctrl-Alt-Backspace` to kill the X server and restart it with `startx -- -nocursor`.
- If you need a terminal session, you can switch to one of the other consoles by pressing `Ctrl-Alt-F2` (or any other function key). Pressing `Ctrl-Alt-F1` brings you back to the first console where Chromium is running.