(/)

Search for... (e.g. Raspberry Pi, Adafruit, Python)

Getting Started with Inky pHAT

Note that this tutorial has been updated since the Inky Python library (https://github.com/pimoroni/inky), which adds support for the larger Inky wHAT display, was introduced. The instructions below will not work with the old Inky pHAT Python library!

Also note that the old Inky pHAT library (https://github.com/pimoroni/inky-phat) is still there if you want to use it, although we strongly suggest you switch to using the new Inky library instead!

This tutorial will show you how to install the Inky Python library (https://github.com/pimoroni/inky), and then walk through its functionality. You'll learn how to run the beautiful included examples: the name badge, calendar, and weather station, and then learn how to display text and images on Inky pHAT.

You'll see that your Inky pHAT already has an image on it, straight out of the bag! This is because e-paper displays, like the one on Inky pHAT, allow you to update them with an image and the image will persist even once you've cut the power supply (i.e. switched off your Pi, or even removed the pHAT altogether)! This means that you can use Inky pHAT as a fancy-pants name badge, completely *sans* power.

Because Inky pHAT comes fully-assembled, there's no need to solder anything, so you can just pop it onto the GPIO pins on your Pi / Pi Zero / W and get going!

Installing the software

Note that if you've previously installed the Inky pHAT Python library and you're installing the Inky library they will live happily alongside each other, so there's no need to get rid of the old library.

We always recommend using the most up-to-date version of Raspbian, as this is what we test our boards and software against, and it often helps to start with a completely fresh install of Raspbian, although this isn't necessary.

As with most of our boards, we've created a really quick and easy one-line-installer to get your Inky pHAT set up. We'd suggest that you use this method to install the Inky pHAT software.

Open a new terminal, and type the following, making sure to type `y` or `n` when prompted:

```
curl https://get.pimoroni.com/inky | bash
```

Once that's done, it probably a good idea to reboot your Pi to let the changes propagate, if the installer doesn't prompt you to reboot.

Using the software

Running the built-in examples

The Inky Python library comes with a handful of really beautiful examples. The examples, that live in the `/home/pi/Pimoroni/inky/examples` folder, are divided into universal ones that work with both the pHAT and wHAT in the topmost folder, and then those that are specific to the pHAT or wHAT in the `phat` and `what` folders respectively.

We'll look at how to run one of the universal examples on the pHAT now, the name badge example. This example takes some command line arguments that specify the type of display (pHAT, in this case) and colour of display that you're using. Command line arguments are extra pieces of information that you can pass into programs straight from the command line.

Assuming you used the one-line-installer to install the library, the examples will have been downloaded to `/home/pi/Pimoroni/inky/examples` (if you cloned and installed the library straight from GitHub then they will be in `inky/examples`). Open a new terminal window, and type the following to navigate to the examples folder:

```
cd /home/pi/Pimoroni/inky/examples
```

Name badge example

Let's try the name badge example. It takes three arguments - `--type`, `--colour`, and `--name` - the names of which should be fairly self-explanatory.

In the terminal, type the following:

```
python name-badge.py --type "phat" --colour "red" --name "Inigo Montoya"
```

(You can also use `-t`, `-c`, and `-n`)

It'll take a few seconds (around 15 usually) to update the whole display. e-paper displays work by pulling coloured particles up and down using different voltages, so that's what all the pulsing of the display is (and why the image persists).

The above assumes that you're using the red/black/white Inky pHAT. If you're using a yellow/black/white or a black/white Inky pHAT then change `"red"` to `"yellow"` or to `"black"` accordingly. And obviously change your name from "Inigo Montoya", unless that really is your name?!

As we said earlier, you can now completely detach your Inky pHAT and use it as a completely powerless name badge, although it should be noted that the image may fade gradually over the course of a day or so, but you can easily perk it up again by reconnecting it to your Pi and running the script again.

Calendar example

We'll look now at the calendar example. This example draws a neat little calendar on your Inky pHAT, with days from the current month in white, overhanging days from the previous and next month in red, and the month and year to the side.

This example is specific to the pHAT but it does take one argument, to specify the colour of Inky pHAT display being used.

In the terminal, type the following to run the calendar example:

```
cd phat
python calendar-phat.py --colour "red"
```

Again, we're assuming that you're using a red/black/white Inky pHAT, but if you're using a yellow/black/white or a black/white one then change the `--colour` to `"yellow"` or `"black"`.

Note that if you want to have the calendar always on and updated then you would have to re-run the script periodically using `crontab`, as follows. In the terminal, type:

```
crontab -e
```

And then, at the very bottom of the file, add the following line:

```
*/30 * * * * python /home/pi/Pimoroni/inky/examples/phat/calendar-phat.py --colour "red"
```

That will run and update the calendar every 30 minutes, although you can change that number to a smaller or larger number depending on how frequently you'd like it to update.

Weather example

Next, we'll take a look at the weather example. It displays the time and date at which the weather was last updated, a little icon describing the current weather, and the temperature and pressure at your chosen location.

You can set your chosen location by changing the `CITY` and `COUNTRYCODE` variables in the `weather-phat.py` script. The defaults are set to `"Sheffield"` and `"GB"`, for obvious reasons...

To run the example, type the following in the terminal:

```
python weather-phat.py --colour "red"
```

The same goes for different colours of Inky pHAT display again...

And again, if you want the example to run periodically and keep the weather updated on your Inky pHAT, then you'll need to use `crontab`, as in the example above. A good frequency at which to run this one is probably every ten minutes.

Building your own code

Let's take a look now at how to build your own code with Inky pHAT. Because of the way that the new universal Inky library works now, there's some boilerplate (yadda yadda code!) that we need at the top of any code we're going to run.

In the terminal, type `python` to open a Python prompt.

Here's the boilerplate. Type it in line by line.

```
from inky import InkyPHAT

inky_display = InkyPHAT("red")
inky_display.set_border(inky_display.WHITE)
```

That code imports the `InkyPHAT` class from the `inky` library, creates an instance of the class that we've called `inky_display` and to which we've passed the argument `"red"` to tell it what colour our display is (change it if you have a different coloured Inky pHAT), and set the border colour (the thin bit at the very edge of the display) to white.

Now we're ready to start displaying things on Inky pHAT!

Displaying text on Inky pHAT

A common task that you might want to do is to display text on Inky pHAT. You can use the Python Image Library (PIL) to display text, using regular TrueType fonts. In fact, we'll be using PIL to display images and graphics on Inky pHAT too.

We've made a Python fonts library to make it easy to use Open Font License (OFL) fonts with our products that have displays. The ones that the Inky library examples use will have be installed as part of the Inky library install.

You can also use fonts the regular way by downloading or transferring them to your Pi and then using the path to the file.

We're going to display a simple `Hello, World!` on Inky pHAT using the Fredoka One font. It's worth noting that we've found that font sizes greater than 22 point look really crisp, anything above about 16 point is legible, and anything less than 16 starts to look less legible, although this will vary depending on the font.

The Python Image Library (PIL) will have been installed when you ran the installer, so there's no need to worry about installing it. Our boilerplate code above has already set up what we need to write to the Inky pHAT display itself, but we'll need to import and set up PIL now.

Type the following:

```
from PIL import Image, ImageFont, ImageDraw

img = Image.new("P", (inky_display.WIDTH, inky_display.HEIGHT))
draw = ImageDraw.Draw(img)
```

This imports three classes from PIL that we'll need, creates a new image, `img` , that is the width and height of the Inky pHAT display, and then creates a drawing canvas, `draw` , to which we can draw text and graphics.

Next, let's import the font we need, and create a variable called `font` that we can use when we're writing text to the canvas.

```
from font_fredoka_one import FredokaOne

font = ImageFont.truetype(FredokaOne, 22)
```

If you want to use your own fonts, then simply replace `FredokaOne` above with the path to your font file in quotes.

As we saw above, when we created the new image, there are handy `inky_display.WIDTH` and `inky_display.HEIGHT` constants that tell us the width and height of the Inky pHAT display, and we can get PIL to tell us the width and height of our `Hello, World!` text, so that we can perfectly centre the text on the display with a little bit of maths!

```
message = "Hello, World!"
w, h = font.getsize(message)
x = (inky_display.WIDTH / 2) - (w / 2)
y = (inky_display.HEIGHT / 2) - (h / 2)
```

The `x` and `y` variables will tell the `draw.text()` function where to place the top left corner of our text. We'll also have to tell the function what colour we want the text (`RED` , `BLACK` , or `WHITE`), and pass it our `font` variable. Last of all, we'll set the image with `inky_display.set_image(img)` and call the `inky_display.show()` function to tell Inky pHAT to refresh the display with our text.

```
draw.text((x, y), message, inky_display.RED, font)
inky_display.set_image(img)
inky_display.show()
```

Try experimenting with different text colours, fonts, and sizes!

Displaying images on Inky pHAT

Displaying images on Inky pHAT requires a little bit of jiggery-pokery to prepare your images properly. They should be PNG-8 images, 212x104 pixels, and in indexed colour mode with a palette of just three colours - black, white, and red - in exactly that order.

We'll run through how to prepare a simple image for Inky pHAT in the free graphics package for Linux, GIMP.

First, in the terminal, we'll install GIMP by typing `sudo apt-get install gimp` and then open it by typing `gimp`.

Go to the `File` menu and click `New` to create a new file. Make the image width 212 pixels and the image height 104 pixels.

Draw your picture. We went for a simple rectangle and circle in black and red.

Once you've finished drawing your picture, you'll need to change the colour palette of the image to a three colour, indexed colour palette image, with the colours in the order white, black, red.

Go to the `Image` menu, then `Mode`, and select `Indexed`.

We've created an Inky colour palette that you can use. In the terminal, type `git clone https://github.com/pimoroni/inky` to clone the GitHub repo. You'll find the colour palette at `inky/tools/inky-palette.gpl`.

Select `Use custom palette`, then click the `Palette selection dialogue` button. Select `Palette file` and then select the `inky-palette.gpl` palette file from the GitHub repo that we just cloned.

The last thing to do is to export the image as a PNG. Go to the `File` menu and then select `Export as`. Give your file a filename (we called ours `inky.png`) and save it in your home directory, `/home/pi`. A dialogue box should pop up with the options for saving it. Make sure that you check the `Save background colour` checkbox, then click `Export`.

Now we have to display our image on Inky pHAT. If you haven't made an image, you can try displaying the Inky pHAT logo file that's in the GitHub repo at `inky/examples/phat/resources/InkyPhat-212x104.png`.

In the terminal, type the following, remembering that you'll have to type the boilerplate for the Inky library and PIL again if you left the Python prompt, and replacing the filename with the name of your own image file if it's different to ours:

```
img = Image.open("/home/pi/inky.png")
inky_display.set_image(img)
inky_display.show()
```

Your Inky pHAT should now be displaying your glorious art.

Taking it further

There's a bunch of drawing tools in the Python Image Library, that you can use to draw on Inky pHAT using code, and without the need for GIMP and indexed colour palettes.

Why not try to write a script that converts regular coloured images to red, black, and white images using Python Image Library? Inky pHAT is ideal for displaying data on. You could use the Matplotlib library to plot out sensor data from other remote Pis with sensors, for example. Or you could hook it up to IFTTT and use it as a tiny message board on which you could leave messages for your family, attached to your fridge perhaps?

Shopping basket

Need something for this project? You can use the links below to add products to your Pimoroni Shop (http://shop.pimoroni.com/) basket for easy checkout.

 Inky pHAT (ePaper/eInk/EPD) (http://shop.pimoroni.com/products/inky-phat) Red/Black/White **£22.50**	Add
 Inky pHAT (ePaper/eInk/EPD) (http://shop.pimoroni.com/products/inky-phat) Yellow/Black/White **£22.50**	Add
 Inky pHAT (ePaper/eInk/EPD) (http://shop.pimoroni.com/products/inky-phat) Black/White **£19.50**	Add
Raspberry Pi Zero (http://shop.pimoroni.com/products/raspberry-pi-zero) **£4.80**	Add
Raspberry Pi Zero W (http://shop.pimoroni.com/products/raspberry-pi-zero-w) **£9.30**	Add
Raspberry Pi 4 (http://shop.pimoroni.com/products/raspberry-pi-4) 4GB RAM **£54.00**	Add
Raspberry Pi 4 (http://shop.pimoroni.com/products/raspberry-pi-4) 2GB RAM **£33.90**	Add
 Official Raspberry Pi Universal Power Supply (Pi 3 & Zero Only) (http://shop.pimoroni.com/products/raspberry-pi-universal-power-supply) **£8.10**	Add
 Universal USB-C Power Supply - 5.1V 3A (http://shop.pimoroni.com/products/universal-usb-c-power-supply-5-1v-3a) **£9.00**	Add
 NOOBS 32GB microSD card (3.3) (http://shop.pimoroni.com/products/noobs-32gb-microsd-card-3-1) **£9.00**	Add
 Antex XS25 Soldering Iron (http://shop.pimoroni.com/products/antex-xs25-soldering-iron-uk-plug) UK Plug **£30.00**	Add
 Antex XS25 Soldering Iron (http://shop.pimoroni.com/products/antex-xs25-soldering-iron-uk-plug) EU plug **£30.00**	Add
Antex Lead Free Solder 2m (http://shop.pimoroni.com/products/copper-solder) **£3.00**	Add
 GPIO Hammer Header (Solderless) (http://shop.pimoroni.com/products/gpio-hammer-header) Male + Female + Installation Jig **£6.00**	Add

- GPIO Hammer Header (Solderless) (http://shop.pimoroni.com/products/gpio-hammer-header)
  Male **£2.10** — Add
- GPIO Hammer Header (Solderless) (http://shop.pimoroni.com/products/gpio-hammer-header)
  Female **£3.00** — Add

Want to checkout or change something? Click here to view your cart (http://shop.pimoroni.com/cart).

Tutorial

Beginner

Raspberry Pi (/?tag=raspberry-pi), Pi Zero (/?tag=pi-zero)

Sandy Macdonald

sandy@pimoroni.com (mailto:sandy@pimoroni.com)
@sandyjmacdonald (https://twitter.com/sandyjmacdonald)
http://sandyjmacdonald.github.io (http://sandyjmacdonald.github.io)

Formerly, Sandy worked at the University of York, in the biology department, analysing data and telling people that they should have used more replicates. Now a fully-fledged crew member at Pimoroni - head of digital content - working on learning materials and digital chunterings. Find him on Twitter and most everywhere else, as sandyjmacdonald.

Our Picade (https://www.kickstarter.com/projects/pimoroni/picade-the-arcade-cabinet-kit-for-your-raspberry-p) was the UKs 1st Kickstarter.
We're the UKs biggest Adafruit (http://shop.pimoroni.com/collections/adafruit) outlet.

Pimoroni Ltd. 2 Manton Street, Sheffield, S2 4BA, United Kingdom.