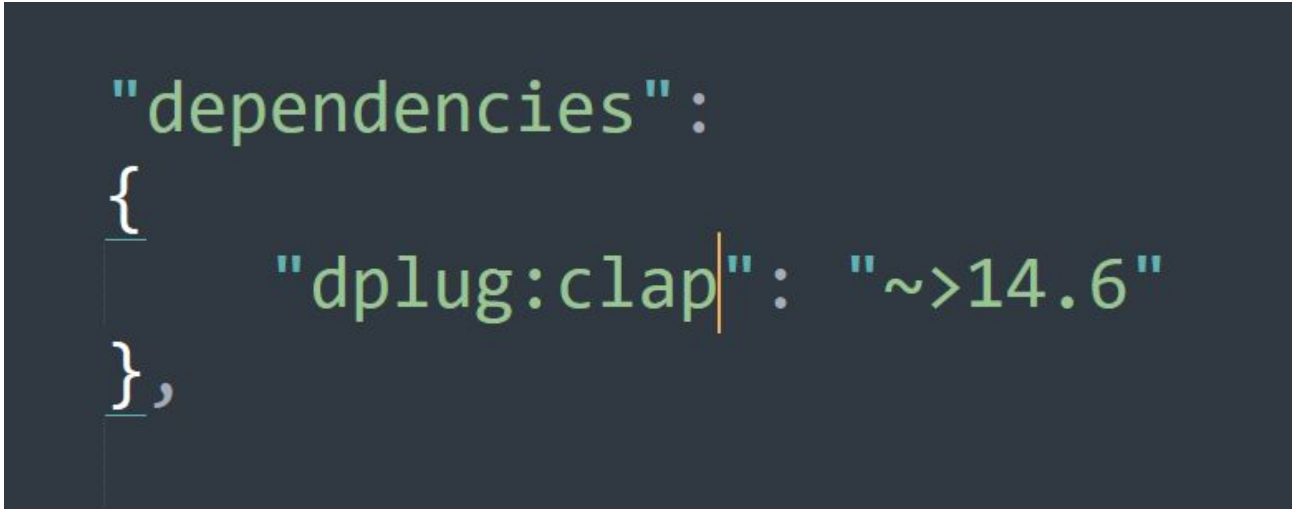
CLAP Support in Dplug and 3 other goodies

Meeting
Nov 19th 2024

1. CLAP support added in Dplug v14.6

STEP ONE Add the package dependency.

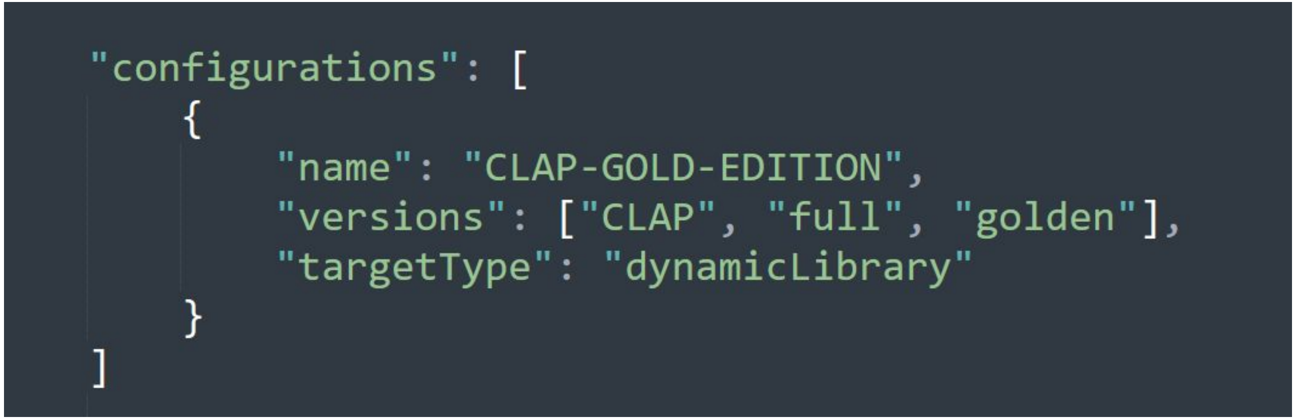
dub.json



```
"dependencies":
{
    "dplug:clap": "~>14.6"
},
```

STEP TWO “CLAP” version identifier

dub.json



```
"configurations": [
    {
        "name": "CLAP-GOLD-EDITION",
        "versions": ["CLAP", "full", "golden"],
        "targetType": "dynamicLibrary"
    }
]
```

```
$ dplug-build -c CLAP-GOLD-EDITION
```

Cool CLAP things

- Active community, good standard

Cool CLAP things

- Active community, good standard

- Neat semantics that can be *stolen* to make a CLAP host (future dplug:host?)
  CLAP is designed to host all other formats.

Cool CLAP things

- Active community, good standard

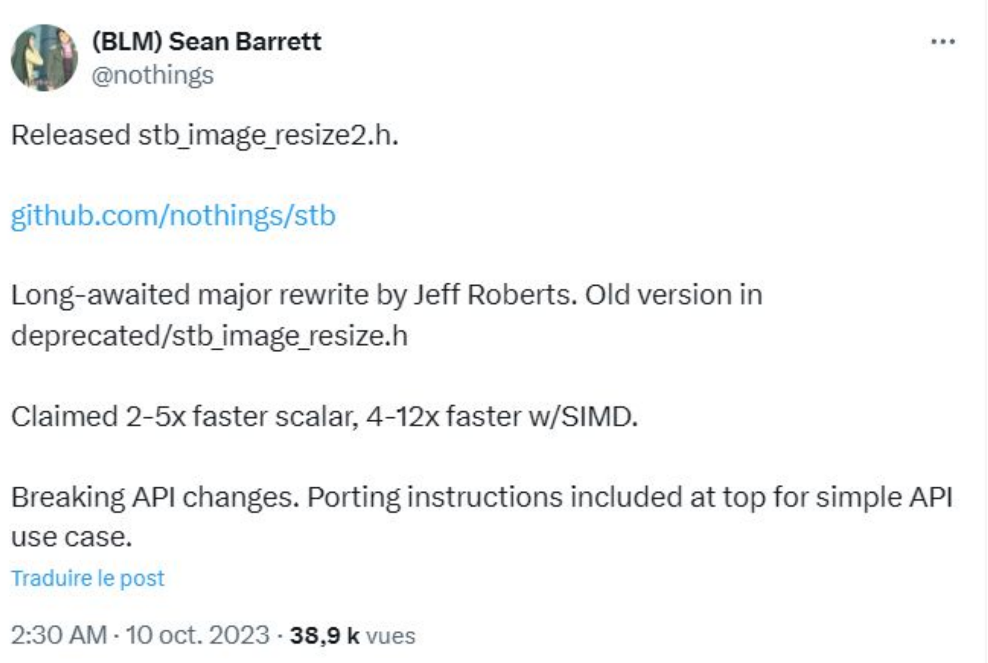
- Neat semantics that can be *stolen* to make a CLAP host (future dplug:host?)
  CLAP is designed to host all other formats. 🤔

- Support got a lot larger in 2024 with Studio One and FL Studio

Cool CLAP things

- Active community, good standard
- Neat semantics that can be *stolen* to make a CLAP host (future dplug:host?)
  CLAP is designed to host all other formats.
- Support got a lot larger in 2024 with Studio One and FL Studio
- Need your help testing it on macOS / Linux!

stb_image_resize2.h



(BLM) Sean Barrett
@nothings

Released stb_image_resize2.h.

github.com/nothings/stb

Long-awaited major rewrite by Jeff Roberts. Old version in deprecated/stb_image_resize.h

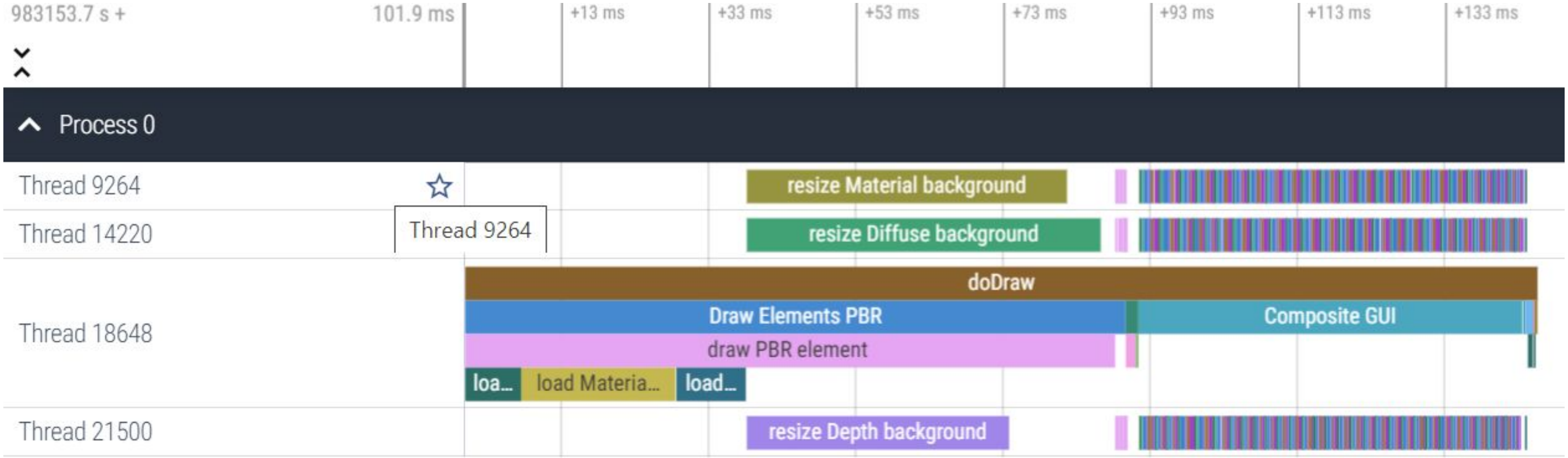
Claimed 2-5x faster scalar, 4-12x faster w/SIMD.

Breaking API changes. Porting instructions included at top for simple API use case.

Traduire le post

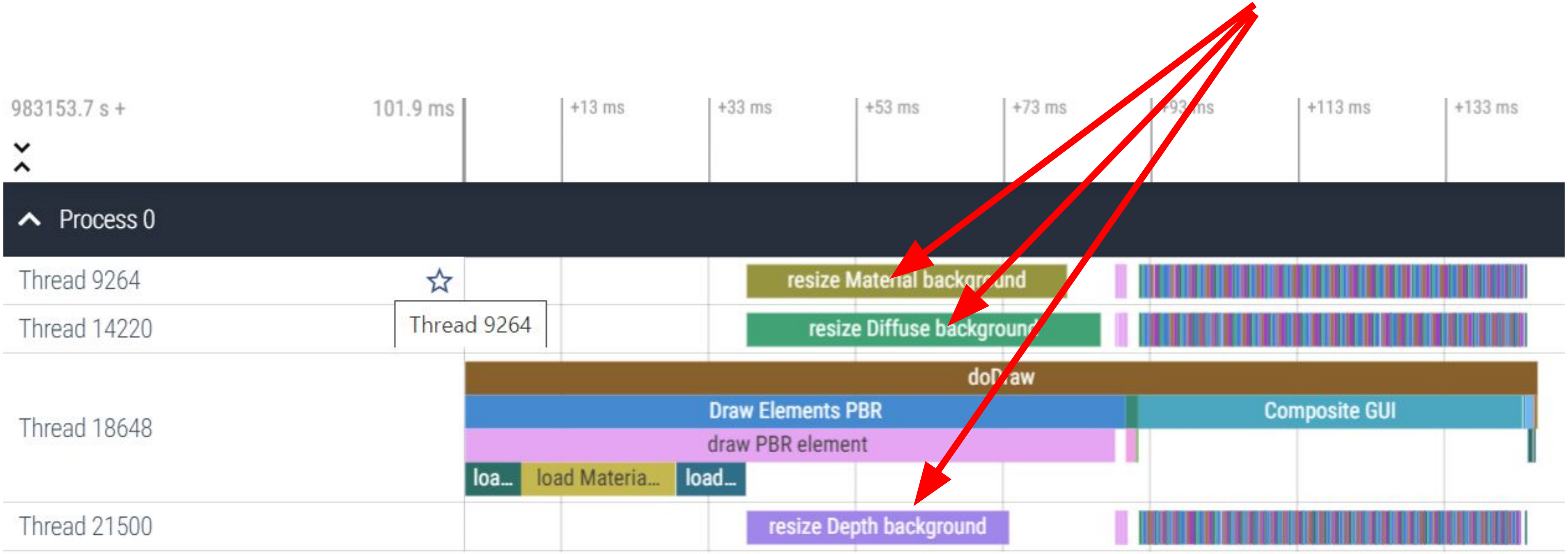
2:30 AM · 10 oct. 2023 · **38,9 k** vues

2. `stb_image_resize2.h` ported to D



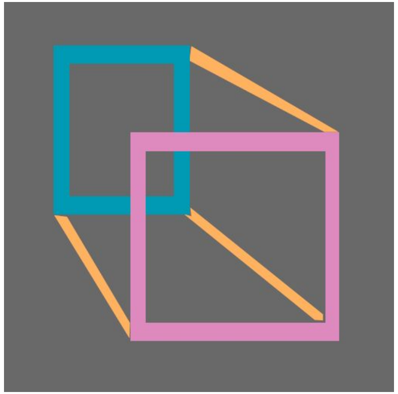
Typical PBR plugin opening.

Every resize now 2x faster



Visibly faster on resize too.

Numbers for `stb_image_resize2-d`



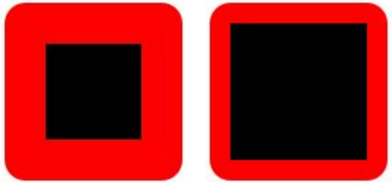
- **200** **mpps**
- **150 kb** binary size
- Lowmem ~ **10%** of image size
- Better quality with Costella's "Magic" kernels:
  a. STBIR_FILTER_MKS_2013_86
  b. STBIR_FILTER_MKS_2021 kernel
- **2.2x** faster than former Dplug resizer

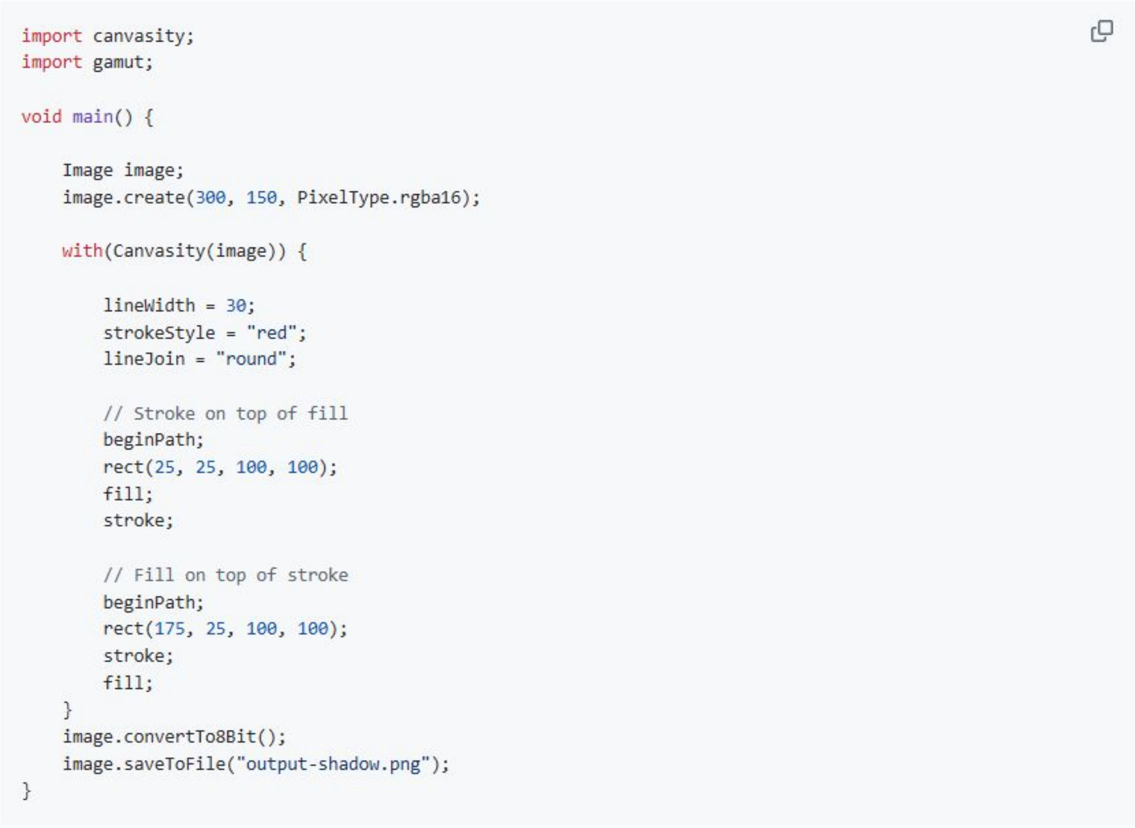
3. `canvas_ity.h` port 🖼️

- New canvas API with **same API** but **different features.**

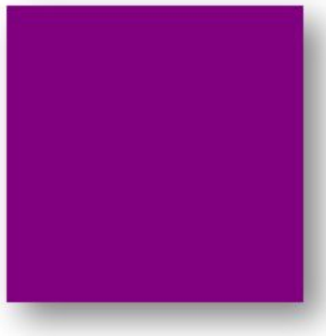
- Slower than `dplug:canvas`, does more things.

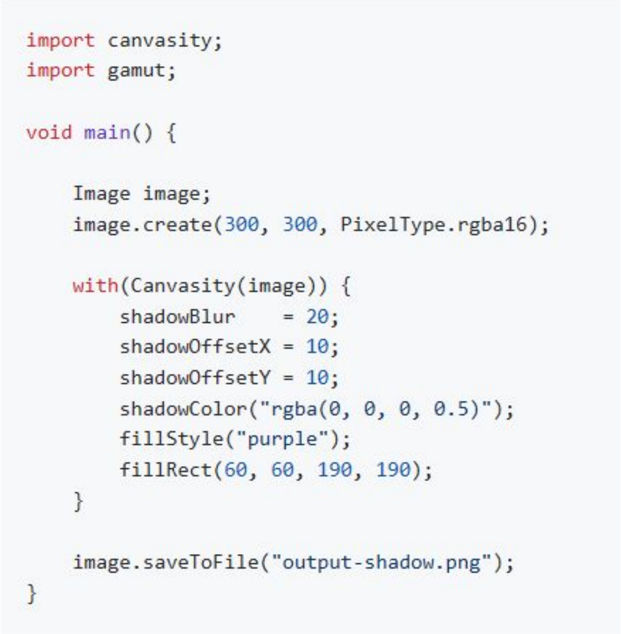
.stroke() and .shadowBlur()



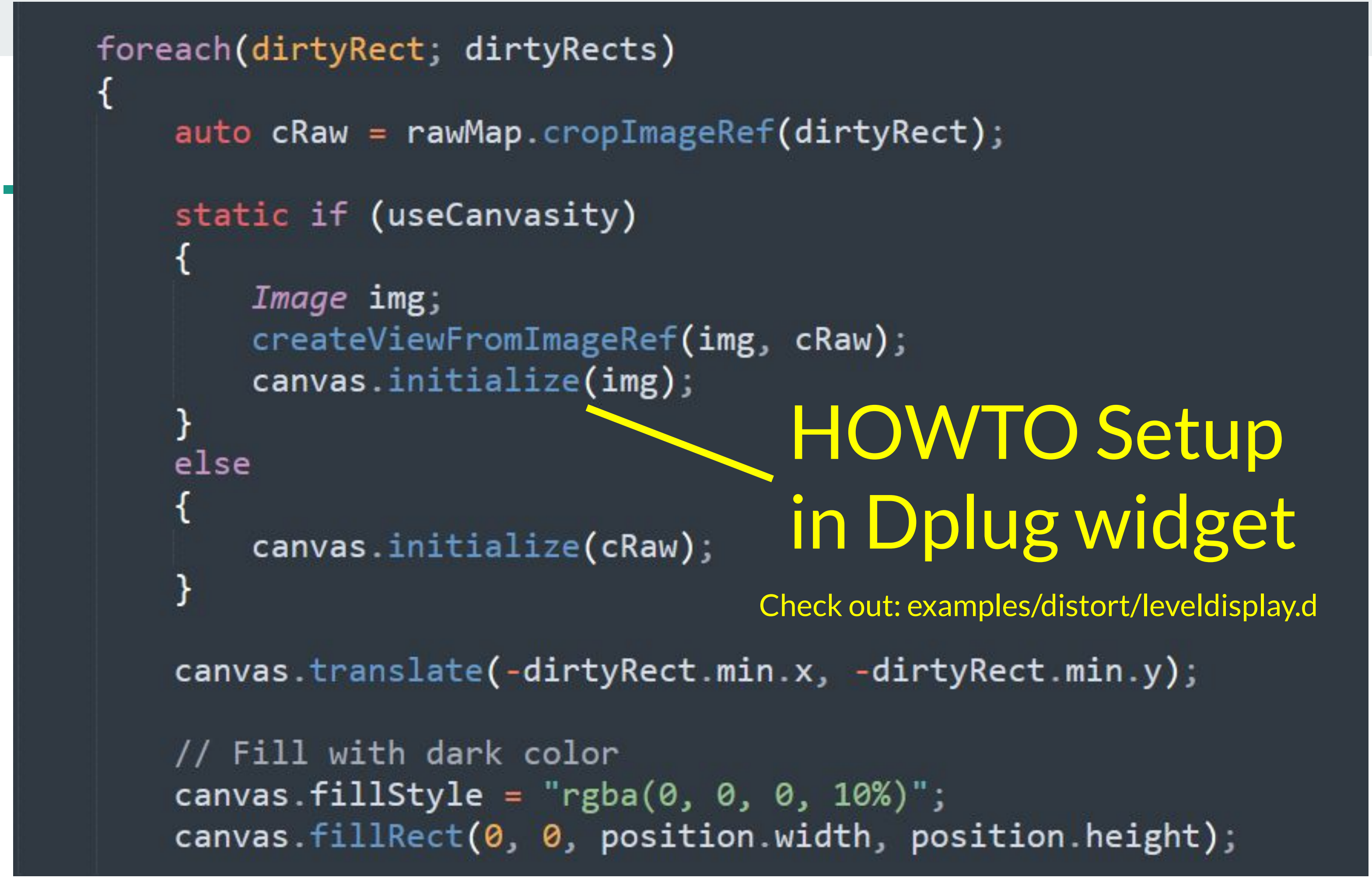


```
import canvasity;
import gamut;

void main() {

    Image image;
    image.create(300, 150, PixelType.rgba16);

    with(Canvasity(image)) {

        lineWidth = 30;
        strokeStyle = "red";
        lineJoin = "round";

        // Stroke on top of fill
        beginPath;
        rect(25, 25, 100, 100);
        fill;
        stroke;

        // Fill on top of stroke
        beginPath;
        rect(175, 25, 100, 100);
        stroke;
        fill;
    }
    image.convertTo8Bit();
    image.saveToFile("output-shadow.png");
}
```





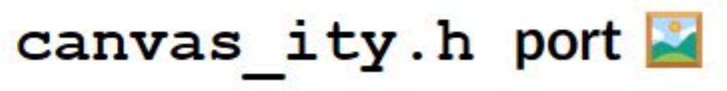
```
import canvasity;
import gamut;

void main() {

    Image image;
    image.create(300, 300, PixelType.rgba16);

    with(Canvasity(image)) {
        shadowBlur    = 20;
        shadowOffsetX = 10;
        shadowOffsetY = 10;
        shadowColor("rgba(0, 0, 0, 0.5)");
        fillStyle("purple");
        fillRect(60, 60, 190, 190);
    }

    image.saveToFile("output-shadow.png");
}
```



```
foreach(dirtyRect; dirtyRects)
{
    auto cRaw = rawMap.cropImageRef(dirtyRect);

    static if (useCanvasity)
    {
        Image img;
        createViewFromImageRef(img, cRaw);
        canvas.initialize(img);
    }
    else
    {
        canvas.initialize(cRaw);
    }

    canvas.translate(-dirtyRect.min.x, -dirtyRect.min.y);

    // Fill with dark color
    canvas.fillStyle = "rgba(0, 0, 0, 10%)";
    canvas.fillRect(0, 0, position.width, position.height);
```

HOWTO Setup in Dplug widget

Check out: examples/distort/leveldisplay.d



canvas_ity.h port 🖼️

Absolutely **un-missable** Wiki page
https://github.com/AuburnSounds/Dplug/wiki/Using-dplug-canvas-or-canvasity

4. `dplug-build --redub` and `dplug-build --reggae`

STEP ONE Install reggae and ninja in your `PATH`

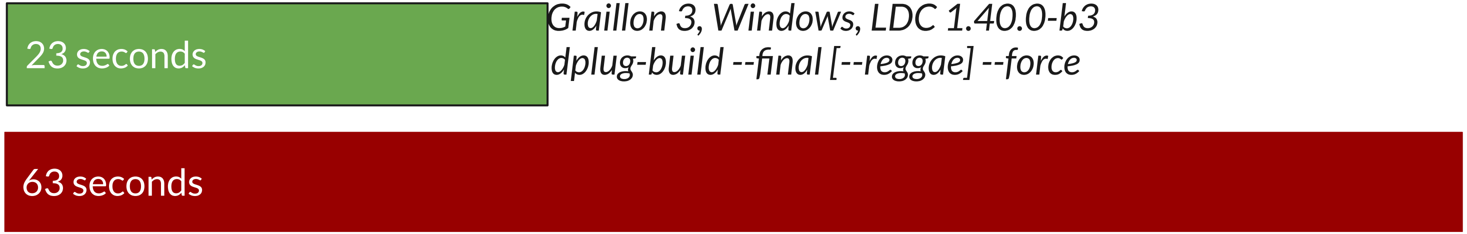
STEP TWO Build recent `dplug-build`

STEP THREE Use `dplug-build --reggae --final` for faster release builds.

(same for redub)

reggae+ninja is pretty fast (-62% time)

- **23 sec** instead of **63 sec** for a FULL OPTIMIZED BUILD!
- **`reggae`** = fast optimized builds, **`redub`** = fast debug builds



Graillon 3, Windows, LDC 1.40.0-b3
dplug-build --final [--reggae] --force

Roadmap 🛣️

- Fix issues and bugs to get back to 10:1 issue closed ratio.
- Remake Dplug website, something more flashy and suitable
- **Dplug v15:** Abandon 32-bit Windows finally

Questions?