QRnZine

A Toolkit for Creating Zines That Contain the Means of their Own Reproduction

An "artistic" exercise in abusing QR codes as a method for memetic self-replication

Preface/Disclaimer

This zine and QRnZine itself are pretty technical in nature and assume that the user is fairly technically advanced going in, with at least basic familiarity with command-line tools and editing of programming, scripting, and markup languages.

The intent of this zine is mostly to explore how a few interesting tools work together to make this document possible, but skips over a lot in the process to fit the medium.

Introduction

Link rot is very real and photocopiers don't come with an easy edit button.

That's basically the problem QRnZine is intended to fix.

If you could put the source code of a zine within the zine itself, other people could hopefully more easily replicate, modify, and create zines built using that one as a template.

But how do you fit the source code for a zine inside itself?

SVGs

First we'll need to have a template to start with...

SVG (Scalable Vector Graphics) is an XML-based vector image format.

If you already know HTML, SVG is super easy to wrap your head around and start modifying, but comes with some handy extra tools for making things like print documents like this one.

SVG seems like a good choice to use for building a document template.

QR Codes

Getting code onto and off of paper easily is a pain if you're relying on manually copying between the two.

Fortunately, this problem is largely avoided using barcodes and 2d barcodes like QR Codes, the ubiquitous 2d barcode solution that pretty much everyone has a scanner for in their pocket.

The biggest QR codes can store about 3kB of text, which is decent, but not quite enough for us.

Data URIs

Data URIs are a special URL format that allows for data to be stored directly in the URL itself, without any backing server.

They usually look like:

```
data:text/html;base64,...
data:image/png;base64,...
```

and are often embedded within webpages, but can be used by themselves and can contain more-or-less any file format, including (quite handily) zip files.

Putting It All Together

By putting SVG source into zip files, placing zip files into data URIs, putting data URIs into QR codes, and referencing the QR codes in the SVG source, we can magically store the source code of our zine in the zine itself more-or-less losslessly.

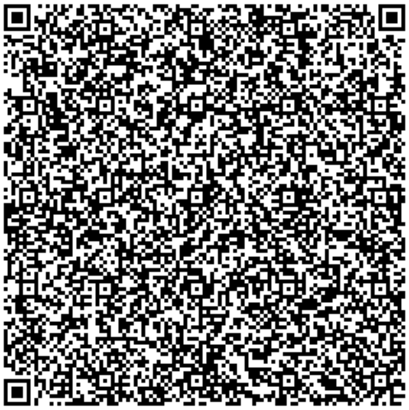
Some caveats apply, like needing software to process SVGs, QR codes, and the automation scripts, but those are hopefully less fragile than a difficult-to-edit PDF with a broken link.

This zine was made with **QRnZine**, a collection of SVG templates and scripts for building zines which contain the means of their own reproduction.

By **unfolding this zine and turning the page over**, you should find a set of QR codes containing instructions and source code that can be used both to **replicate this zine**, as well as to **create your own zines** with their own self-replicating abilities.

https://github.com/mlaga97/QRnZine
QRnZine v0.2

The QR code immediately to the right has a data URI to a zip file containing a README.md with more info. (Nesting doll approach results in huge space savings)

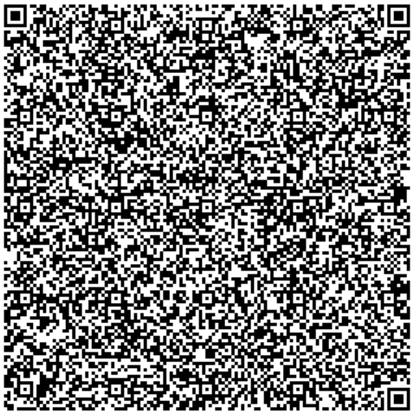


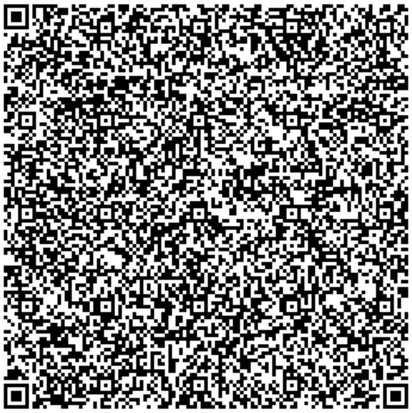
Instructions: Scan each of the QR codes on this page, pasting the data URI contained within each into a browser URL bar in order to download a set of zip files containing the individual files listed next to each QR code.

When all of the files are combined together into a single directory on a Linux computer, they can be used as-is to recreate the entirety of this zine, or modified to create a new zine that also has "self-replicating" abilities.

This Zine was made using QRnZine v0.2
Upstream Source Code: https://github.com/mlaga97/QRnZine

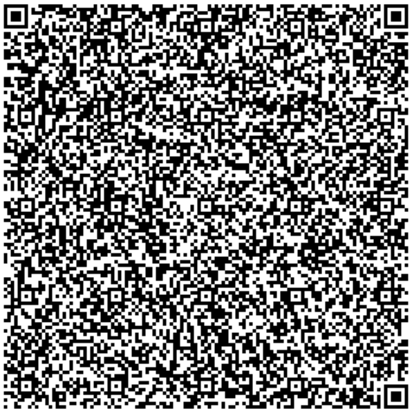
./main.svg
./build.sh

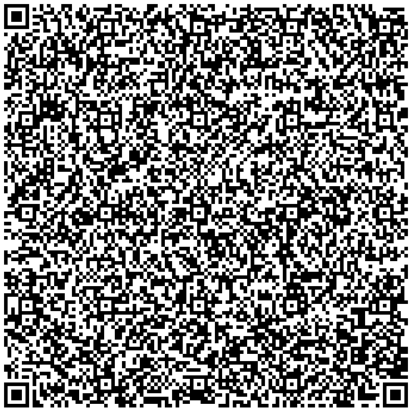




./c4.svg

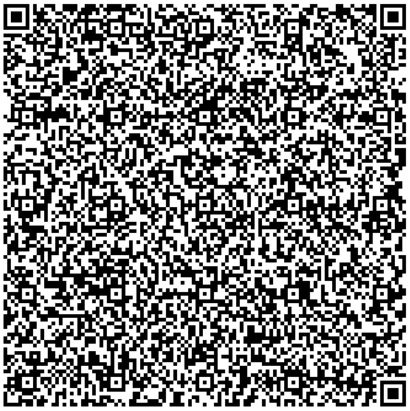
./instructions.svg
./clean.sh

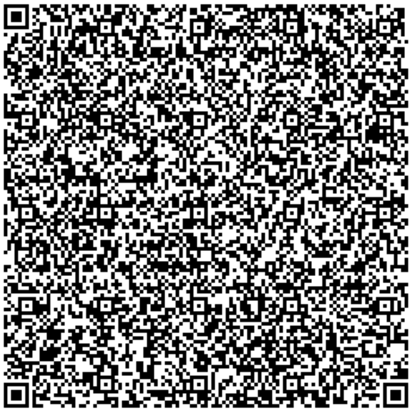




./c3.svg

./c1.svg





./c2.svg