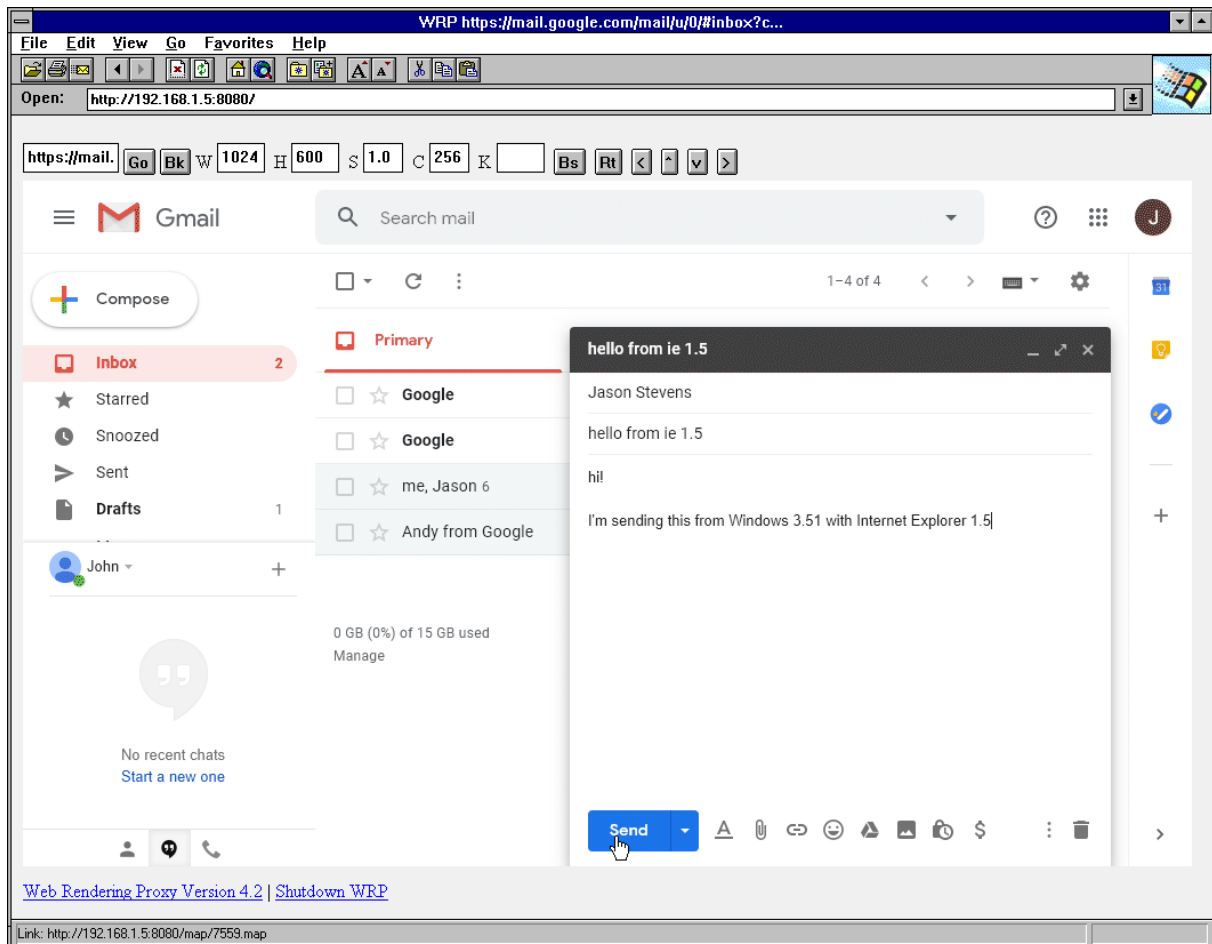

WRP - Web Rendering Proxy

A browser-in-browser “proxy” server that allows to use historical / vintage web browsers on the modern web. It works by rendering a web page in to a GIF or PNG image with clickable imagemap.



Usage Instructions

- Download a WRP binary and run it on a machine that will become your WRP gateway/server. This should be modern hardware, OS and Google Chrome / Chromium Browser is required to be preinstalled. Do not try to run WRP on an old machine like Windows XP or 98.
- Make sure you have disabled firewall or open port WRP is listening on (by default 8080).
- Point your legacy browser to <http://address:port> of the WRP server. Do not set or use it as a “proxy server”.
- Type a search string or a full http/https URL and click **Go**.
- Adjust your screen **Width/Height/Scale/Colors** to fit in your old browser.

-
- Scroll web page by clicking on the in-image scroll bar.
 - WRP also allows **a single tall image without the vertical scrollbar** and use client scrolling. To enable this, simply height **H** to 0 . However this should not be used with old and low spec clients. Such tall images will be very large, take a lot of memory and long time to process, especially for GIFs.
 - Do not use client browser history-back, instead use **Bk** button in the app.
 - You can re-capture page screenshot without reloading by using **St** (Stop). This is useful if page didn't render fully before screenshot is taken.
 - You can also reload and re-capture current page with **Re** (Reload).
 - To send keystrokes, fill **K** input box and press **Go**. There also are buttons for backspace, enter and arrow keys.
 - Prefer PNG over GIF if your browser supports it. PNG is much faster, whereas GIF requires a lot of additional processing on both client and server to encode/decode. Jpeg encoding is also quite fast.
 - GIF images are by default encoded with 216 colors, "web safe" palette. This uses an ultra fast but not very accurate color mapping algorithm. If you want better color representation switch to 256 color mode.

UI explanation

The first unnamed input box is either search (google) or URL starting with http/https

Go instructs browser to navigate to the url or perform search

Bk is History Back

St is Stop, also re-capture screenshot without refreshing page, for example if page render takes a long time or it changes periodically

Re is Reload

W is width in pixels, adjust it to get rid of horizontal scroll bar

H is height in pixels, adjust it to get rid of vertical scroll bar. It can also be set to 0 to produce one very tall image and use client scroll. This 0 size is experimental, buggy and should be used with PNG and lots of memory on a client side.

Z is zoom or scale

C is colors, for GIF images only (unused in PNG, JPG)

K is keystroke input, you can type some letters in it and when you click Go it will be typed in the remote browser.

Bs is backspace

Rt is return / enter

< ^ v > are arrow keys, typically for navigating a map, buggy.

UI Customization

WRP supports customizing it's own UI using HTML Template file. Download wrp.html place in the same directory with wrp binary customize it to your liking.

Docker

```
1 $ docker run -d -p 80:8080 tenox7/wrp
```

Google Cloud Run

```
1 $ gcloud run deploy --platform managed --image=gcr.io/tenox7/wrp:latest
   --memory=2Gi --args='-t=png', '-g=1280x0x256'
```

Or from Gcloud Console. Use `gcr.io/tenox7/wrp:latest` as container image URL.

Note that unfortunately GCR forces https. Your browser support of encryption protocols and certification authorities will vary.

Azure Container Instances

```
1 $ az container create --resource-group wrp --name wrp --image gcr.io/
   tenox7/wrp:latest --cpu 1 --memory 2 --ports 80 --protocol tcp --os-
   type Linux --ip-address Public --command-line '/wrp -l :80 -t png -g
   1280x0x256'
```

Or from the Azure Console. Use `gcr.io/tenox7/wrp:latest` or `tenox7/wrp:latest` for image name.

Fortunately ACI allows port 80 without encryption.

Flags

```
1 -l listen address:port (default :8080)
2 -t image type gif, png or jpg (default gif)
3 -g image geometry, WxHxC, height can be 0 for unlimited (default 1152
  x600x216)
4     C (number of colors) is only used for GIF
5 -q Jpeg image quality, default 80%
6 -h headless mode, hide browser window on the server (default true)
7 -d chromedp debug logging (default false)
8 -n do not free maps and images after use (default false)
9 -ui html template file (default "wrp.html")
10 -ua user agent, override the default "headless" agent
11 -s delay/sleep after page is rendered before screenshot is taken (
    default 2s)
```

Minimal Requirements

- Server/Gateway requires modern hardware and operating system that is supported by Go language and Chrome/Chromium Browser, which must be installed.
- Client Browser needs to support [HTML FORMs](#) and [ISMAP](#). Typically Mosaic 2.0 would be minimum version for forms. However ISMAP was supported since 0.6B, so if you manually enter url using `?url=...`, you can use the earlier version.

Troubleshooting

I can't get it to run

This program does not have a GUI and is run from the command line. After downloading, you may need to enable executable bit on Unix systems, for example:

```
1 $ cd ~/Downloads
2 $ chmod +x wrp-amd64-macos
3 $ ./wrp-amd64-macos
```

Websites are blocking headless browsers

This is a well known issue. WRP has some provisions to work around it, but it's a cat and mouse game. The first and foremost recommendation is to change [User Agent](#), so that it doesn't say "headless". Add `-ua="my agent"` to override the default one. Obtain your regular desktop browser user agent and specify it as the flag. For example

```
1 $ wrp -ua="Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36"
```

History

- Version 1.0 (2014) started as a *cgi-bin* script, adaptation of [webkit2png.py](#) and [pcidade.py](#), blog post.
- Version 2.0 became a stand alone http-proxy server, supporting both Linux and MacOS, another post.
- In 2016 thanks to EFF/Certbot the whole internet migrated to HTTPS/SSL/TLS and WRP largely stopped working. Python code became unmaintainable and there was no easy way to make it work on Windows, even under WSL.
- Version 3.0 (2019) has been rewritten in Go using Chromedp as browser-in-browser instead of http-proxy. The initial version was less than 100 lines of code.
- Version 4.0 has been completely refactored to use mouse clicks via imagemap instead parsing a href nodes.
- Version 4.1 added sending keystrokes in to input boxes. You can now login to Gmail. Also now runs as a Docker container and on Cloud Run/Azure Containers.
- Version 4.5 introduces rendering whole pages in to a single tall image with client scrolling.
- Version 4.6 adds blazing fast gif encoding by Hill Ma.

Credits

- Uses chromedp, thanks to mvdan for dealing with my issues
- Uses go-quantize, thanks to ericpauley for developing the missing go quantizer
- Thanks to Jason Stevens of Fun With Virtualization for graciously hosting my rumblings
- Thanks to claunia for help with the Python/Webkit version in the past
- Thanks to Hill Ma for ultra fast gif encoding algorithm
- Historical Python/Webkit versions and prior art can be seen in wrp-old repo

Legal Stuff

License: Apache 2.0

Copyright (c) 2013-2018 Antoni Sawicki

Copyright (c) 2019-2024 Google LLC