Using Puppeteer for Automated Web Testing and other Nefarious Activities

Matt Kaufman

About

Matt Kaufman

- Chief Innovation Officer @ MK Partners
- Volunteer Organizer for GDG San Fernando Valley
- Volunteer Mentor for GDG Pacific Region

the1mattkaufman

Repo: https://www.github.com/the1mattkaufman/puppeteer-service

PDF of these slides are in the repo

Prereqs for this Intermediate Talk

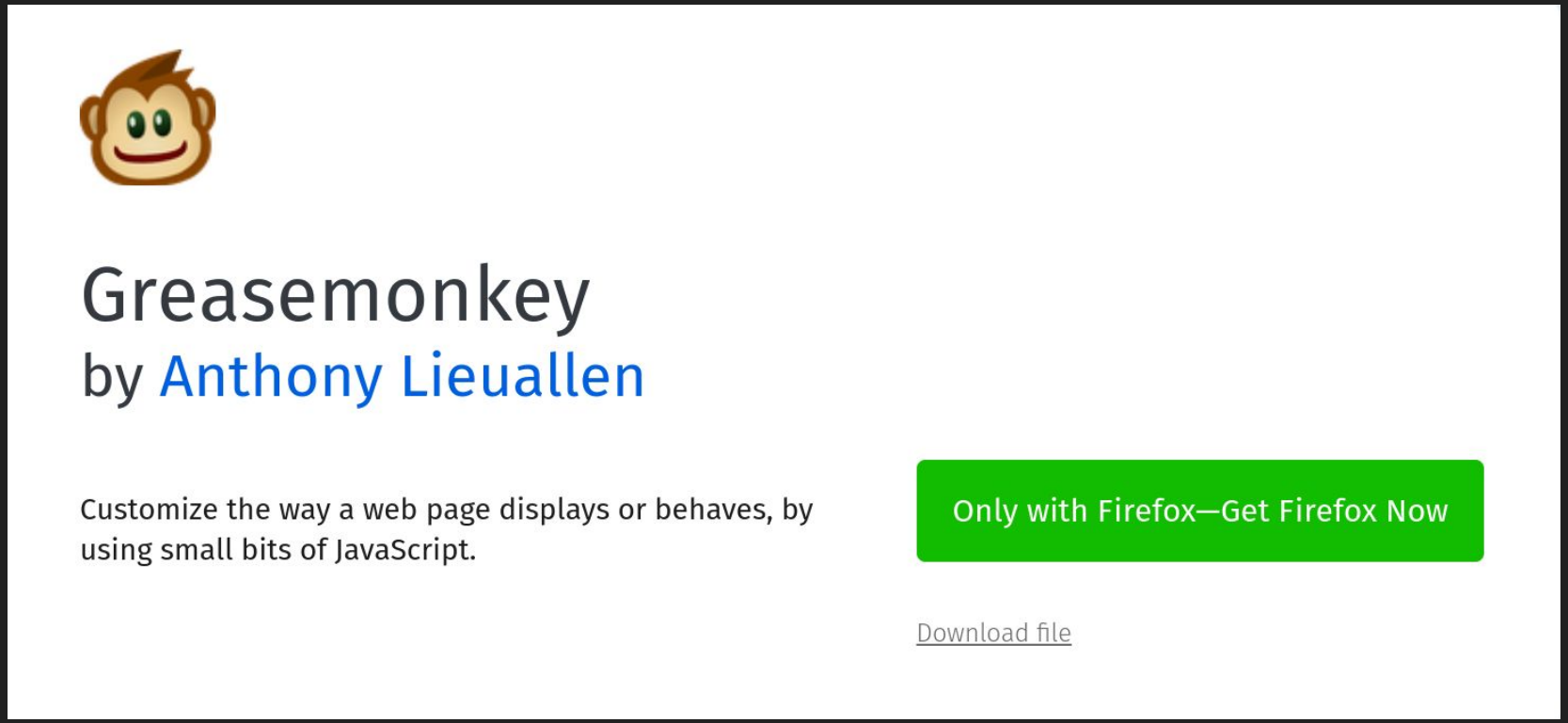
- JavaScript (required)
- HTML/CSS (required)
- nodeJS (required)
- GitHub (recommended)
- Docker (optional)
- Google Cloud Platform (optional)
- Google Cloud Build (optional)
- Google Cloud Run (optional)

I have a problem

I need to navigate to a URL multiple times a day and

1. Make sure it’s still live and functioning
2. Take a screenshot of it
3. Scrape some data
4. Fill out and submit a form

Original Browser Automation Option



Sample GreaseMonkey Script

```
function clickEverything(){
 setTimeout(function(){ clickClassName("youtube")}, 500);
 setTimeout(function(){ clickClassName("screenshot")}, 20000 );
 setTimeout(function(){ clickId("divcontent")},30000);
 setTimeout(function(){
window.location.assign("https://www.myurl.com") }, 1200000 );
}
```

Better Browser Automation Option

Puppeteer

build error | npm v5.5.0 | issue resolution 21 d

API | FAQ | Contributing | Troubleshooting

> Puppeteer is a Node library which provides a high-level API to control Chrome or Chromium over the DevTools Protocol. Puppeteer runs headless by default, but can be configured to run full (non-headless) Chrome or Chromium.

What can I do?

https://github.com/puppeteer/puppeteer

How to get started

Install nodeJS

https://nodejs.org/en/download/

install puppeteer

```
npm i puppeteer
```

Build a node app

Ugh, seriously?

How about we try before we “buy”

Puppeteer Sandbox
1.19.0

1. basics - Emulate devices

Run Save Embed

```
1 const puppeteer = require('puppeteer')
2 const iPhone = puppeteer.devices['iPhone 6'];
3
4
5 const browser = await puppeteer.launch()
6 const page = await browser.newPage()
7 await page.emulate(iPhone)
8 await page.goto('https://google.com/')
9 await page.screenshot({
10     path: 'full.png',
11     fullPage: true
12 })
13 console.log(await page.title())
14 await browser.close()
```

Console output

Nothing here yet

Pick one of the examples and run it!

https://try-puppeteer.appspot.com/

https://puppeteersandbox.com/

Our puppeteer app

```
const puppeteer = require("puppeteer");
const browser = await puppeteer.launch();
const page = await browser.newPage().catch((e) => {
await page.goto(url);
//do stuff here
await page.close()
```

What kind of stuff can we do now?

- Go to a URL
- Wait for x ms / an element to be rendered / no network activity, etc
- Navigate the DOM
- Click on Elements
- Interact with Inputs
- Interact with alerts / confirm / popup
- Take a screenshot
- Execute JS on the page
- Use JS in our app to control resulting behavior

Let’s put our functionality into a node app

```
const express = require("express");
const app = express();
const server = app.listen(process.env.PORT || 8080, (err) => {
 if (err) return console.error(err);
 const port = server.address().port;
});
app.use(async (req, res) => {
    //your code goes here
}
module.exports = app;
```

Where should I host my app?

Low entry cost

Secure

Reliable

Scalable

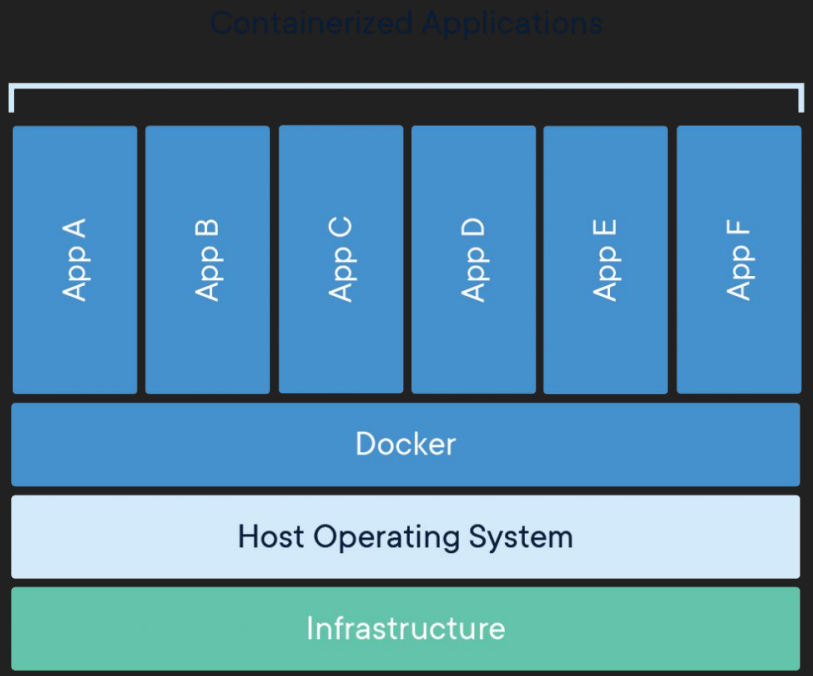
Serverless

Well-documented

Container !== ‘Scary’

A Container consists of your code and a everything required to run it.

Containers allow you to scale or migrate to new infrastructure with exact consistency



Docker Containerize your App

Dockerfile

Tells Docker Engine what to load on run

.dockerignore

Tells Docker what not to include in your container image

Getting started with Cloud Run

1. Create Project in Google Cloud Console
2. Enable Billing
3. Enable APIs
4. Update IAM permissions
5. Install and initialize Cloud SDK locally
6. Containerize your app
7. Build container image
8. Deploy container image to Cloud Run

Let’s Build our Container Image

```
# Use Cloud Build to build the current directory as a Google Container Image
gcloud builds submit --tag gcr.io/PROJECT-ID/APP-NAME

# View the Image in the Container Registry
https://console.cloud.google.com/gcr/images/PROJECT-ID/GLOBAL/APP-NAME
```

Let’s Deploy our Image to Cloud Run

```
# Deploy an Image to Cloud Run
gcloud run deploy --image gcr.io/PROJECT-ID/APP-NAME --platform
managed

# Set Region
# Allow unauthenticated invocations to allow access to the public

# View our service running in Cloud Run
https://console.cloud.google.com/run?project=PROJECT-ID
```

Shortcut - Cloud Code extension for VS Code

Cloud Code

Create a Kubernetes Sample App | Create a Cloud Run Sample App | Release Notes

The tools you need for Cloud Native development leveraging your favorite OS, IDE, language and cloud.

Key Features

- Support for Go, Node, Java, and Python
- Rapid Edit, Package, Run loop to your K8s cluster
- Integrated debugging and log viewing/streaming
- Snippets, completions, and linting for K8s artifacts
- Cluster management, resource browsing, and K8s cluster inspection
- Cluster creation support for Amazon EKS, Azure AKS and Google GKE
- Support for Custom Resources (CRDs) e.g. Istio, Knative
- Deploy, service browsing and log viewing support for Cloud Run fully managed and Cloud Run for Anthos
- Managing Cloud API and Libraries
- Support for Buildpacks
- Automatically uses your Google Cloud SDK credentials
- Google Secret Manager integration: Create secrets in the IDE and access them from your code to keep your apps secure.

Resources

Browse our docs
We have a lot of features to discover! Head over to our documentation to read more about them.

Join us on Slack
Connect with the Cloud Code team in the #cloud-code Slack channel.

File an issue
If you discover an issue, please file a bug and we'll fix it ASAP.

Request a feature
If you have any feature requests, ideas for improvement, or general feedback; please submit a feature request.

Prerequisites

The Power of Cloud Build

Cloud Build does more than build container images from local directories

- It integrates with github, bitbucket, and Google Cloud Repositories
- It is aware of changes made to your repos
- It lets you define triggers based on changes
- It lets you define actions to take when a trigger is invoked
- It has a simple dashboard

Build Configuration file

Steps that should be executed as part of your build process.

Can reference 3rd party libraries/packages/images

Can run tests

Can fail intentionally or due to issues

Scheduling with Cloud Scheduler

Fully managed, enterprise-grade scheduler

Cloud Scheduler is a fully managed enterprise-grade cron job scheduler. It allows you to schedule virtually any job, including batch, big data jobs, cloud infrastructure operations, and more. You can automate everything, including retries in case of failure to reduce manual toil and intervention. Cloud Scheduler even acts as a single pane of glass, allowing you to manage all your automation tasks from one place.

It’s Demo Time!

Further Learning

https://pptr.dev/

https://github.com/berstend/puppeteer-extra/tree/master/packages

https://cloud.google.com/run/docs

https://docs.docker.com/

https://cloud.google.com/cloud-build/docs

https://cloud.google.com/scheduler