Tutorial Use Puppeteer on node.js in Armbian (for web automation)

By schwar3kat
August 11, 2019 in Reviews, Tutorials, Hardware hacks

schwar3kat
Posted August 11, 2019

How to install Puppeteer on node.js on Armbian. Tested on OrangePi Zero Plus running Ubuntu Xenial or Bionic.
 Should work on other boards and other builds.

How to install the latest version of node.js. (at the time of writing was 12.7.0).
Instructions for installing node.js on Debian and Ubuntu can be found at:
https://github.com/nodesource/distributions/blob/master/README.md#deb

The instructions at the time of writing were:

# Using Ubuntu

```
curl -sL https://deb.nodesource.com/setup_12.x | sudo -E bash -
sudo apt-get install -y nodejs
```

# Using Debian, as root

```
curl -sL https://deb.nodesource.com/setup_12.x | bash -
apt-get install -y nodejs
```

Both of these methods work on Ubuntu.

If you then try to install Puppeteer using "npm i puppeteer" you will discover that the version of Chrome that it installs locally is for the wrong CPU architecture.
The solution to the problem is to install and use puppeteer-core instead of puppeteer.
When using puppeteer core, you must define the path to Chrome or Chromium in the puppeteer script (see the test example below).

First install a recent version of Chromium Browser. This will be used in headerless mode.

```
apt-get install chromium-browser
```

When the chromium-browser install is complete
Create a folder for your Puppeteer project

Let's assume that your folder is in /mnt/data/nodejs-scripts/screenshot (mine was on a mounted ntfs drive).

```
cd /mnt/data/nodejs-scripts/screenshot

npm init -y

npm i puppeteer-core --save
```

Now create a node.js test script
Lets say that you call it screenshot.js
Using your favorite editor insert the following lines into the file.

```
const puppeteer = require('puppeteer-core');

(async ()=>{
    const browser = await puppeteer.launch({
        executablePath: '/usr/bin/chromium-browser',
        args: ['--no-sandbox'],
        headless: true
    });

const page = await browser.newPage();

await page.goto('https://duckduckgo.com/');

await page.screenshot({path: 'screenshot.png'});

await browser.close();

})();
```

Make the script executable.

```
chmod +x screenshot.js
```

You can now run the script

```
cd /mnt/data/nodejs-scripts/screenshot

nodejs screenshot.js
```

If it has worked, you will find a screenshot of the DuckDuckGo page in your project folder.

This topic is now closed to further replies.

Share Followers 0

Go to topic listing