Node.js: how to create a screenshot of a web page with Puppeteer on desktop and Linux servers

In this tutorial we will see how to use Puppeteer in Node.js to create screenshots of web pages on both desktop and Linux servers.

The key aspect to understand is that Puppeteer does nothing but provides an API that creates and manages an instance of a browser installed locally.

No special requirements are needed on the desktop, while on Linux servers the necessary dependencies must first be installed to allow the browser to operate.

So first on Linux servers we need to install these dependencies.

```
# Debian/Ubuntu

sudo apt-get update

sudo apt-get install -y gconf-service libasound2
libatk1.0-0 libatk-bridge2.0-0 libc6 libcairo2
libcups2 libdbus-1-3 libexpat1 libfontconfig1 libgcc1
libgconf-2-4 libgdk-pixbuf2.0-0 libglib2.0-0 libgtk-
3-0 libnspr4 libpango-1.0-0 libpangocairo-1.0-0
libstdc++6 libx11-6 libx11-xcb1 libxcb1
libxcomposite1 libxcursor1 libxdamage1 libxext6
libxfixes3 libxi6 libxrandr2 libxrender1 libxss1
libxtst6 ca-certificates fonts-liberation
```

```
libappindicator1 libnss3 lsb-release xdg-utils wget

sudo apt-get install -y libgbm-dev
```

Another difference between desktop and Linux servers is in the creation of the browser instance via the Puppeteer API. On Linux servers, the `launch()` method must be invoked as follows:

```
const browser = await puppeteer.launch({args: ['--no-sandbox']});
```

What we want to implement is an API endpoint that creates a screenshot from the URL and the size passed by the client with a POST request and returns the data URL of the newly created screenshot.

We can then create the following class:

```
'use strict';

const puppeteer = require('puppeteer');
const validator = require('validator');

class Preview {
    constructor(url, width, height) {
        this.url = url;
        this.width = width;
        this.height = height;
    }

    validate() {
        if (!validator.isURL(this.url)) {
            return false;
        }
        if (this.width < 300 || this.width > 2560) {
```

```
            return false;
        }
        if (this.height < 500 || this.height > 1700)
{
            return false;
        }
        return true;
    }

    async create() {
        if (!this.validate()) {
            return Promise.reject();
        }
        try {
            const browser = await puppeteer.launch();
            // Server: puppeteer.launch({args: ['--
no-sandbox']});
            const page = await browser.newPage();
            await page.setViewport({
                width: this.width,
                height: this.height,
                deviceScaleFactor: 1
            });
            await page.goto(this.url);
            const data = await
page.screenshot({encoding: 'base64'});
            await browser.close();
            return `${data}`;

        } catch (err) {
            console.log(err);
            return Promise.reject(err);
        }
    }
```

```
}

module.exports = Preview;
```

Our endpoint will therefore be:

```
'use strict';

const express = require('express');
const router = express.Router();
const Preview = require('../classes/Preview');

router.post('/screenshot', async (req, res, next) =>
{
    let url = '', width = 0, height = 0;
    if(req.body.url && req.body.width &&
req.body.height) {
        url = req.body.url;
        width = req.body.width;
        height = req.body.height;
    }
    const preview = new Preview(url, width, height);
    try {
        const image = await preview.create();
        res.send({ image });
    } catch(error) {
        res.send({ error  });
    }
});
```

As you can see, once the dependencies on the server are installed, the actual procedure is relatively straightforward.