Self-contained, working example (thanks **[@verglor](https://github.com/verglor)**)

const fs = require('fs');

const path = require('path');

const puppeteer = require('puppeteer');

const util = require('util');

// set up, invoke the function, wait for the download to complete

async function download(page, f) {

const downloadPath = path.resolve(

process.cwd(),

`download-${Math.random()

.toString(36)

.substr(2, 8)}`,

);

await util.promisify(fs.mkdir)(downloadPath);

console.error('Download directory:', downloadPath);

await page.\_client.send('Page.setDownloadBehavior', {

behavior: 'allow',

downloadPath: downloadPath,

});

await f();

console.error('Downloading...');

let fileName;

while (!fileName || fileName.endsWith('.crdownload')) {

await new Promise(resolve => setTimeout(resolve, 100));

[fileName] = await util.promisify(fs.readdir)(downloadPath);

}

const filePath = path.resolve(downloadPath, fileName);

console.error('Downloaded file:', filePath);

return filePath;

}

// example usage

(async function() {

const browser = await puppeteer.launch({ headless: true });

try {

const page = await browser.newPage();

await page.goto(

'http://file-examples.com/index.php/text-files-and-archives-download/',

{ waitUntil: 'domcontentloaded' },

);

const path = await download(page, () =>

page.click(

'a[href="http://file-examples.com/wp-content/uploads/2017/02/file\_example\_CSV\_5000.csv"]',

),

);

const { size } = await util.promisify(fs.stat)(path);

console.log(path, `${size}B`);

} finally {

await browser.close();

}

})().catch(e => {

console.error(e.stack);

process.exit(1);

});