

Join GitHub today

GitHub is home to over 31 million developers working together to host and review code, manage projects, and build software together.

Sign up

Dismiss

window inner size not equal to viewport size #1183

New issue

 Closed

chanjsq opened this issue on Oct 27, 2017 · 46 comments



chanjsq commented on Oct 27, 2017

Tell us about your environment:

- Puppeteer version: 0.13.0-alpha
- Platform / OS version: macos high sierra 10.13
- URLs (if applicable):

What steps will reproduce the problem?

Please include code that reproduces the issue.

```
// start.js

const puppeteer = require('puppeteer')

const run = async () => {
  const browser = await puppeteer.launch({
    headless: false,
    slowMo: 250,
    args: [
      '--disable-infobars',
    ],
  });
};

const page = await browser.newPage();

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

await page.waitFor(60000)
await browser.close();
}

run()
```

1. `node start.js`

What is the expected result?

1. window inner size and viewport should be equal, all the default 800px x 600px

What happens instead?

1. window inner size is larger than the viewport size

Assignees

No one assigned

Labels

feature

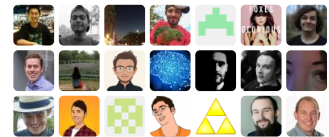
Projects

None yet

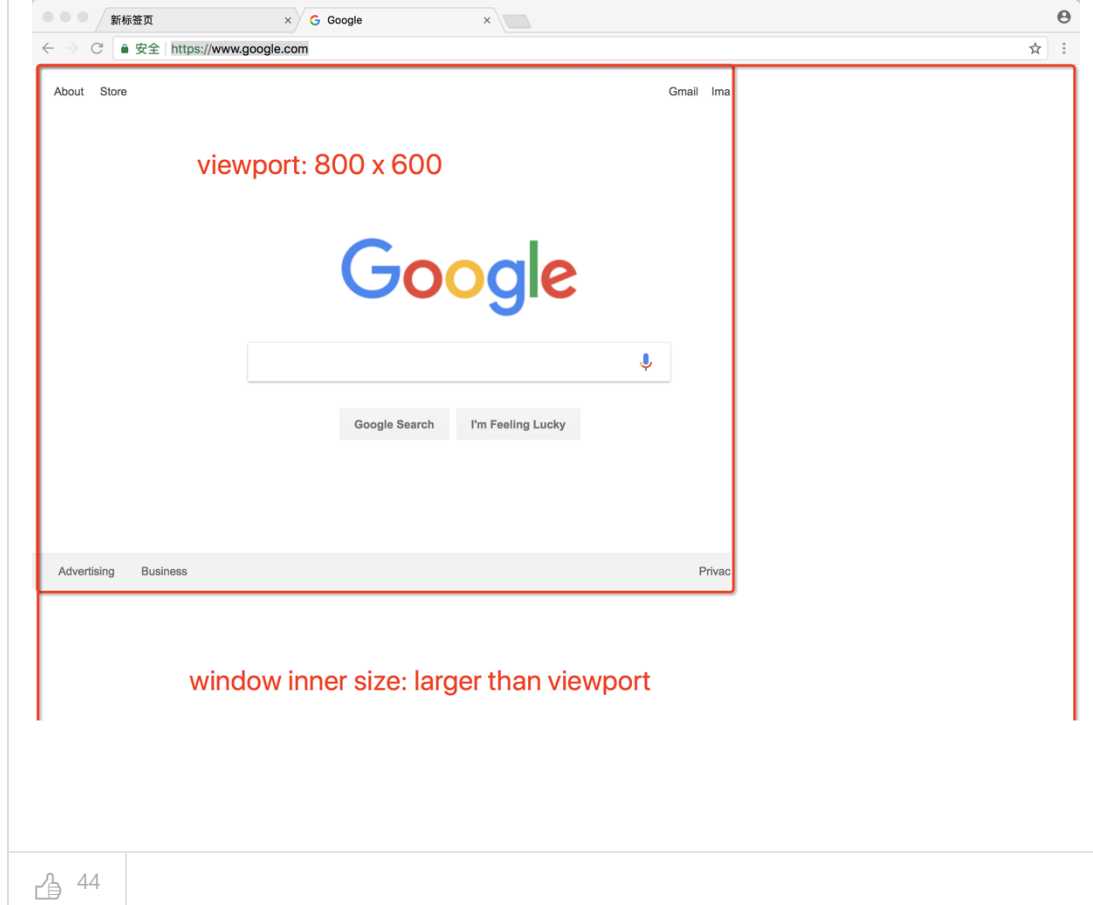
Milestone

No milestone

26 participants



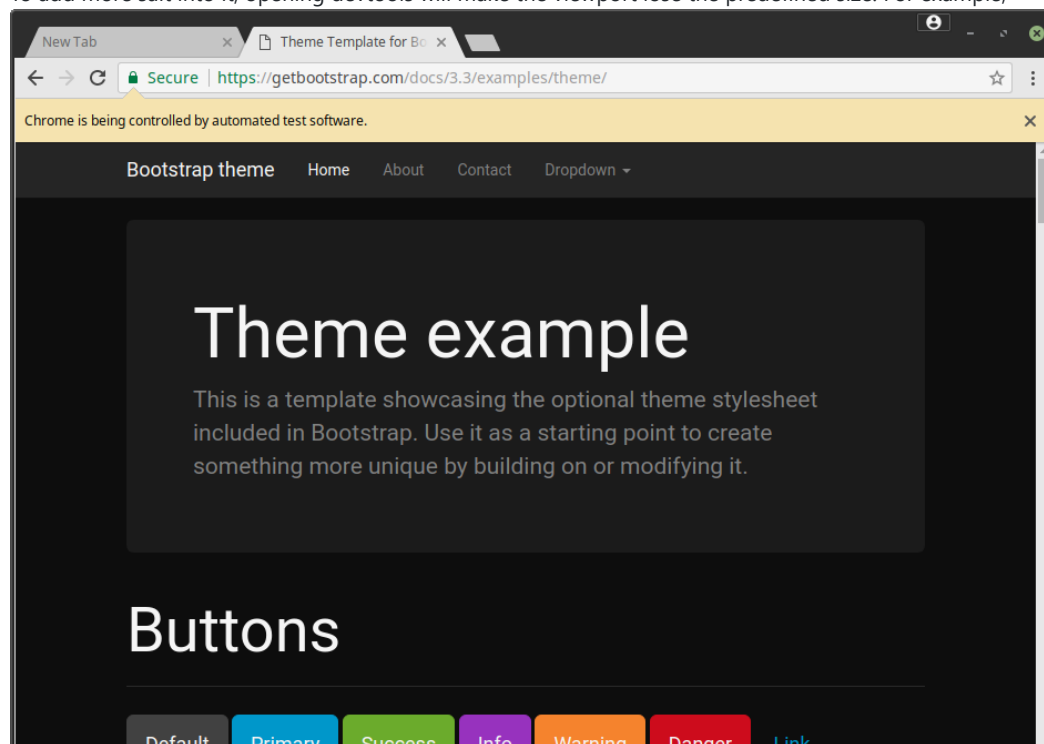
and others



entrptaher commented on Oct 29, 2017 • edited ▾

Contributor

To add more salt into it, opening devtools will make the viewport lose the predefined size. For example,





4



1



zicongxie commented on Oct 29, 2017

It has a param can be reset the window viewports in launch method.

...



chanjsq commented on Oct 29, 2017

Author

@zicongxie `await page.setViewport({ width: 600, height: 400 })` only change the viewport size, but the window size will not changed to 600px x 400px too.



4



1



chanjsq commented on Oct 29, 2017

Author

@entrptaher I noticed that too.



aslushnikov commented on Oct 31, 2017

Contributor

@chanjsq what's your usecase? Why would you need to change the window size?



entrptaher commented on Nov 1, 2017

Contributor

For me, say I will want to test responsiveness.



5



chanjsq commented on Nov 1, 2017 • edited

Author

@aslushnikov When setting the viewport size larger than the default window size, we also need to set the window size with the same value to see the whole page width, it's convenient if the window size can change with viewport size accordingly.



mike-aung-san commented on Nov 1, 2017

Have the same issue on Windows 8.



2



aslushnikov commented on Nov 1, 2017

Contributor

@entrptaher isn't viewport enough for responsiveness?
@chanjsq so this is mostly for debugging purposes?



chanjsq commented on Nov 1, 2017

Author

@aslushnikov yes, smart can be better



z-vr commented on Nov 15, 2017

can i make the viewport equal to the size of the window by default, and adjustable while resizing? i don't want the viewport which is not equal to the window size :(+1 on this one

👍 28

🔖 aslushnikov added the `feature` label on Nov 17, 2017



radekmie commented on Dec 2, 2017

I'm using this code:

```
await page.setViewport({height, width});

// Window frame - probably OS and WM dependent.
height += 85;

// Any tab.
const {targetInfos: [{targetId}]} = await browser._connection.send(
  'Target.getTargets'
);

// Tab window.
const {windowId} = await browser._connection.send(
  'Browser.getWindowForTarget',
  {targetId}
);

// Resize.
await browser._connection.send('Browser.setWindowBounds', {
  bounds: {height, width},
  windowId
});
```

I think that `Tab window` + `Resize` parts could be part `browser` API such as:

```
browser.resize({height, width})
```

👍 26



tnoetzel commented on Dec 14, 2017

Definitely also need this, fwiw.



ganfubin commented on Jan 15, 2018

this function is very necessar



xjrimus commented on Feb 6, 2018

+1



akylkb commented on Feb 7, 2018 • edited ▼

<https://peter.sh/experiments/chromium-command-line-switches/#window-size>

```
const puppeteer = require('puppeteer')

const run = async () => {

  // Viewport && Window size
  const width = 400
  const height = 600

  const browser = await puppeteer.launch({
    headless: false,
    args: [
      `--window-size=${width} ${height}`
    ]
  })
}
```

```

    },
    });

    const page = await browser.newPage();

    // Adaptive?
    // await page.setUserAgent('Mozilla (Android) Mobile')

    await page.setViewport({ width, height })
    await page.goto('https://www.google.com/')

    // await browser.close()
  }

  run()

```

👍 25

Mehuge commented on Feb 10, 2018 • edited ▼

I am using the following to achieve the desired effect.

```

const chrome = { x: 0, y: 74 }; // comes from config in reality
args.push(`--window-size=${width+chrome.x},${height+chrome.y}`);
puppeteer.launch({ headless, sloMo, args });
// ...
page.setViewport({ width, height });

```

👍 6

tomholub commented on Feb 16, 2018

Based on <https://github.com/GoogleChrome/puppeteer/blob/v0.13.0/docs/api.md#pagefocusselector> and @Mehuge asnwer

```

const chrome = { x: 0, y: 74 }; // comes from config in reality
args.push(`--window-size=${width+chrome.x},${height+chrome.y}`);
puppeteer.launch({ headless, sloMo, args });
// ...
page.setViewport({ width, height });

```

(correcting typo in `setViewport` above)

👍 3

I-keep-trying commented on Feb 26, 2018

Thanks, guys!!! I combined @akylkb with @Mehuge and got the perfect solution!

And fwiw, for people who didn't understand why viewport size is important. It was stopping me dead in my tracks, because a common practice in many websites is to change the layout from pc / laptop to mobile based on viewport size, and the default size was causing it to go to the mobile layout, which did not have the login button available without an extra menu, involving two extra clicks. The entire DOM and navigation path can be completely different depending on how many different responsive designs the website designers have in place; they could have several, from smart watches to mobile, to ipads and tablets, touch screens and kiosks as well. The world includes much, much more than just laptops and pcs, and each and every layout will have a completely different DOM structure.

So, again, THANK YOU, @akylkb and @Mehuge, your solutions deserve a lot more attention imo!!

hubidu referenced this issue on Mar 1, 2018

Puppeteer: `resizeWindow()` does not change window bounds #973

🔔 Open

grantstephens commented on Mar 1, 2018

Think this is actually a bug when connecting over a websocket as I can't find a way to set the window size when on web socket. This means that some sites render incorrectly as they seem to look at the window size and not that set viewport, i.e:

Launch the browser

Connect to it via websocket

Set the viewport to be bigger than the window size of the websocket browser

Screenshot is based on window size.

I'm not 100% sure of this, but would seem to be the case and would appreciate anybody who could confirm.



PayteR commented on Mar 19, 2018 • edited ▼

Hi, thx guys for solutions, but `screen.width` and `screen.height` are in `headless: true` always 800x600 and for `headless: false` is always resolution of actual screen. Any suggestions to solve this other than manually do it so in `evaluate()` method? Code:

```
await Page.evaluateOnNewDocument((screenWidth, screenHeight) => {
  screen.__defineGetter__('width', function () {
    return screenWidth;
  });
  screen.__defineGetter__('height', function () {
    return screenHeight;
  });
}, width, height);
```



4

gavoja added a commit to gavoja/CodeceptJS that referenced this issue on Mar 29, 2018

Workaround for GoogleChrome/puppeteer#1183

2dea910

DavertMik added a commit to Codeception/CodeceptJS that referenced this issue on Mar 30, 2018

Workaround for GoogleChrome/puppeteer#1183 (#1007) ...

✓ a58d0c5

yrshaikh referenced this issue on Apr 10, 2018

Neither Page.setViewport() nor paper format can change window width when save as PDF #2333

Closed



winterli commented on Apr 10, 2018

This might be related issue,

When saving as PDF (have to use headless mode), the window size triggers the incorrect media query. For example, if `page.setViewport(1200,960)`, then load below HTML page and save to PDF. Will expect the DIV has a gray background. But it actually shows a yellow background.

```
<body>
  <script>
    document.write("Window width: "+window.innerWidth);
  </script>
  <style>
    div{
      background: gray;
    }
    @media (max-width: 767px) {
      div{
        background: yellow;
      }
    }
  </style>
  <div>gray indicates window > 767px, yellow indicates window <=767px</div>
</body>
```

Window width: 1200

gray indicates window > 767px, yellow indicates window <=767px




1



aslushnikov added a commit to aslushnikov/puppeteer that referenced this issue on Apr 13, 2018



 feat(Page): allow disabling viewport emulation ...

9add1b2



 aslushnikov referenced this issue on Apr 13, 2018

feat(Page): allow disabling viewport emulation #2364

 Closed

 Mehuge commented on Apr 13, 2018 • edited ▾

I am now using @radekmie's solution, as well as the --window-size argument.

In my test environment, the browser size may be specified before the browser window is open, and may be altered afterwards. So in the case where it is specified before the browser window is open, I remember the size and use --window-size when launching the browser. If the test scripts later specify a new size, I then use the resize code provided by @radekmie.



2

2 hidden items

[Load more...](#)



 aslushnikov referenced this issue on Apr 27, 2018

No way to determine native DPR of device #2358

 Closed

 TheCloudlessSky commented on Apr 27, 2018

@Maluen this solution worked for my problem getting `window.devicePixelRatio` in #2372.



 aslushnikov referenced this issue on May 17, 2018

wrong viewport in headful mode by default #2558

 Closed

 DanielRuf commented on May 17, 2018 • edited ▾

Definitely the same issue that I have.

Would make sense to disable the viewport emulation in headful mode.

Or provide an option like in the linked PR.

 winterli commented on May 23, 2018

@Maluen the `Emulation.clearDeviceMetricsOverride` solution seems not working in headless mode which is the only mode support Save to PDF feature :(

 DanielRuf commented on May 23, 2018

@winterli which version of Puppeteer do you use?

Should normally work with this.

<https://github.com/DanielRuf/website-checks/blob/master/index.js#L131>



winterli commented on May 24, 2018

@DanielRuf I'm using v1.4.0 from NPM. Below is the code I'm using,

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

(async () => {
  const browser = await puppeteer.launch({
    headless: true,
    args: [
      '--window-size=1280,960'
    ],
  });
  const page = await browser.newPage();
  await page._client.send('Emulation.clearDeviceMetricsOverride');
  const url = "http://winterli.atwebpages.com/test.html";
  await page.goto(url, {waitUntil: 'networkidle2'});
  await page.pdf({path: 'viewport_test.pdf', format: "A4", printBackground: true});
  await browser.close();
  console.log("done!");
})();
```

Did I miss anything here?



DanielRuf commented on May 24, 2018 • edited ▼



DanielRuf commented on May 24, 2018

In general I do not see any `viewport` metatag. Please add one.



DanielRuf commented on May 24, 2018 • edited ▼

<https://codepen.io/DanielRuf/pen/QrRyvV>



winterli commented on May 24, 2018

@DanielRuf thanks for your response. In the test page, I put some CSS media query to test the viewport. If the viewport is wider than 767, the second line should have a gray background. If the viewport is narrower than 767, the second line should have a yellow background. In my example, the window width is 1280, I should see a gray background for the second line. But as the screenshot you shared shows, it's not the case. PS: I have added the viewport tag in the test page also.



entrptaher commented on May 24, 2018

Contributor

I've tested it out, the max viewport height I could do was around 20000px from my computer. I tried to take a ultra fullpage screenshot :D ...



mbwhite commented on May 29, 2018

@Maluen solution has worked for me just now; also I did discover that if you go into debug model (ctrl-shift-I) that has the required effect. Useful for manual testing etc.



silkcom commented on Jul 20, 2018

I'm having this same problem and it effects the way that css with vw and vh work, since they go based on the window size not the viewport size. It's a problem in both headless and not.



aslushnikov closed this in [2563213](#) on Aug 2, 2018



chanjsq commented on Aug 2, 2018

Author

Awesome!



danielmello commented on Aug 14, 2018

I'm having this same problem with puppeteer@1.7.0. When you launch the browser with defaultViewport null and headless true, nothing change, but with headless false, the viewport is affected.



hegelstad commented on Aug 16, 2018

Surely some strange things still going on.



eliucs referenced this issue on Aug 17, 2018

Chromium starts off with a large gap on the side of the screen #3103

Closed



hermanho referenced this issue on Sep 17, 2018

SetViewportAsync is not working in PDF Generation #576

Closed



junguchina commented on Sep 17, 2018

Same problem, is there any solution to change the options in [puppeteerlaunchoptions](#)? When I set the value like below:

```
defaultViewport: {  
  width: 1920,  
  height: 1080  
}
```

The viewport is still 800*600, this is not convenient to debug by viewport.



junguchina referenced this issue on Sep 17, 2018

Can I change the viewport size same to window inner size with Puppeteer helper?

Open

#1209

DanielRuf commented on Sep 17, 2018



See [2563213](#) @junguchina



DanielRuf commented on Sep 17, 2018

Please also try the latest release.

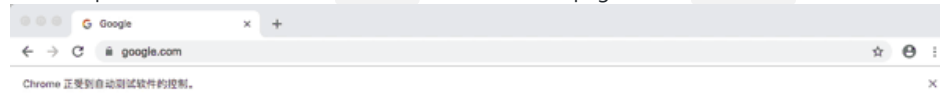


junguchina commented on Sep 17, 2018 • edited ▼

@DanielRuf Thanks, my version of puppeteer is the latest version(1.8.0).

It still has this problem.I use codeceptjs with puppeteer.

The viewport size of chromium is 800x600 ,but inner homepage size is 1920x1080 .



3



fergardi commented on Sep 24, 2018

I'm facing the same issue as above.



DanielRuf commented on Sep 24, 2018

Mh weird, it seems this was not resolved as I am still getting the narrow / small viewport even if I resize. Is there some regression?



fergardi commented on Sep 24, 2018

I want to share my personal experience. This will be a long trip.

I'm using puppeteer@1.8.0 and Google Chrome V69.0.3497.100 in a Windows10 environment.

Let's say I am new to Headless Chrome, and I want to print a dashboard in PDF. I searched for a typical responsive template dashboard and found this one I will be using as an example, credits to the original author: <https://demos.creative-tim.com/bs3/paper-dashboard/dashboard.html>

Then I create a new fresh script to work with:

```
const puppeteer = require('puppeteer');
(async () => {
  const browser = await puppeteer.launch({headless: false}) // debug
  const page = await browser.newPage()
  await page.setViewport({width: 768, height: 1024, deviceScaleFactor: 2})
  await page.goto('https://demos.creative-tim.com/bs3/paper-dashboard/dashboard.html', {waitUntil: 'networkidle0'})
  await page.waitFor(1000)
  await page.click('button.close') // remove notification
  await page.click('a.navbar-brand') // click on page to close advertisement
  await page.emulateMedia('screen') // force screen CSS
  await page.waitFor(1000) // wait for every modal to dissappear
  await page.pdf({path: `${Date.now()}.pdf`, format: 'A4', printBackground: true})
  await browser.close()
})();
```

After that, I prove that in *headful* mode, and it gives me this result:



Firstly, why that blank space? Is that the difference between the default 800px width viewport minus my 768px viewport? I don't think so. However, this does not show in the final PDF, only in *headful* mode, but it stills bugs me a lot. In *headless* mode, it shows at follows:



Secondly, why the media CSS flex is not triggering as showed in the intial picture? There should be two columned layout, not the single one. As I inspect them, they show as classic '@media' queries. Maybe the 'screen' forced mode instead of 'print' is causing this? I do not know.

Thirdly, why my pixel resolution is not being respected? If, in *headful* mode, I show the DevTools and extract them from the page (which seems, by the way, to re-render the page to the expected behaviour), I can see the pixel resolution is not at all the one I setted of in the initial step.

In fact, I could even increase the pixel resolution to have no results on the output:

Ah, that seems the old 800x600 viewport issue we were talking about. So I started to test every solution in this post, with these results:

1. **Force puppeteer to launch to a nulled defaultViewport?** No changes either in headless or headful.
2. **Force puppeteer to launch to a defaultViewport with exactly the same resolution as I wanted?** No changes either in headless or headful.
3. **Force puppeteer to launch to a defined --window-size arg commands?** That seems indeed to change the headful result and fix the strange blank space, leaving me with my desired initial resolution (to test responsiveness, for example). That leads me to the next issue. If I'm visiting this website with 768px width, which effectively triggers the corresponding `@media` query, why in *headful* I see one result and in *headless* I see another different result?

Maybe the float is affecting the behaviour? But again, why am I not achieving the exact same result being in *headful* or in *headless*, as this is the whole point? I do not know. That will be an entirely different issue.



3



2



LarenDorr added a commit to LarenDorr/puppeteer that referenced this issue 10 days ago



feat: add option to specify the default viewport (GoogleChrome#3005) ...

5fc37ec



valentinmagot referenced this issue 18 hours ago

Added resizeScreenSize steps and tests #42

Merged