

Home > puppeteer > Page > waitForFunction

Page.waitForFunction() method

The `waitForFunction` can be used to observe viewport size change:

```
const puppeteer = require('puppeteer');
(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  const watchDog = page.waitForFunction('window.innerWidth < 100');
  await page.setViewport({ width: 50, height: 50 });
  await watchDog;
  await browser.close();
})();
```

To pass arguments from node.js to the predicate of `page.waitForFunction` function:

```
const selector = '.foo';
await page.waitForFunction(
  (selector) => !!document.querySelector(selector),
  {},
  selector
);
```

The predicate of `page.waitForFunction` can be asynchronous too:

```
const username = 'github-username';
await page.waitForFunction(
  async (username) => {
    const githubResponse = await fetch(
      `https://api.github.com/users/${username}`
    );
    const githubUser = await githubResponse.json();
    // show the avatar
    const img = document.createElement('img');
    img.src = githubUser.avatar_url;
    // wait 3 seconds
    await new Promise((resolve, reject) => setTimeout(resolve, 3000));
    img.remove();
  },
  {},
  username
);
```

Signature:

```
waitForFunction(pageFunction: Function | string, options?: {
```

```

    timeout?: number;
    polling?: string | number;
  }, ...args: SerializableOrJSHandle[]): Promise<JSHandle>;

```

Parameters

| Parameter | Type | Description |
|--------------|--|---|
| pageFunction | Function string | Function to be evaluated in browser context |
| options | { timeout?: number; polling?: string number; } | Optional waiting parameters |
| args | SerializableOrJSHandle[] | Arguments to pass to pageFunction |

Returns:

Promise<JSHandle>

Promise which resolves when the **pageFunction** returns a truthy value. It resolves to a JSHandle of the truthy value.

The optional waiting parameter can be:

- **Polling**: An interval at which the **pageFunction** is executed, defaults to **raf**. If **polling** is a number, then it is treated as an interval in milliseconds at which the function would be executed. If polling is a string, then it can be one of the following values: - **raf**: to constantly execute **pageFunction** in **requestAnimationFrame** callback. This is the tightest polling mode which is suitable to observe styling changes. - **mutation**: to execute **pageFunction** on every DOM mutation.
- **timeout**: maximum time to wait for in milliseconds. Defaults to 30000 (30 seconds). Pass 0 to disable timeout. The default value can be changed by using the **page.setDefaultTimeout(timeout)** method.