# **Playwright简介**

Playwright为现代 Web 应用程序提供可靠的端到端测试。

　端到端测试，end-to-end，简称 e2e，也被称作功能测试(Functional Testing)或者浏览器测试或者冒烟测试，是指从使用者的角度出发，对真实系统进行测试，e2e 测试本质上是一种黑盒测试，相当于模拟用户访问应用程序，主要检查界面或功能是否正确，自动化测试不完善的时候通常是由人工来完成这项测试工作。

**支持所有浏览器**

* 在Chromium, Firefox 和 WebKit上运行测试：Google Chrome和Microsoft Edge都是基于Chromium项目，Apple Safari基于WebKit，还有Mozilla 的Firefox。

官网：<https://playwright.dev/>

# **浏览器引擎**

Trident(IE)

IE 1.0 1995

IE 2.0 1995.12

IE 3.0内置到window系统中

WebKit(

-2005年苹果开源了Safari内核Webkit

Safari

Chrome (2008年)

2009(extensions)

2012年，成为最受欢迎的浏览器

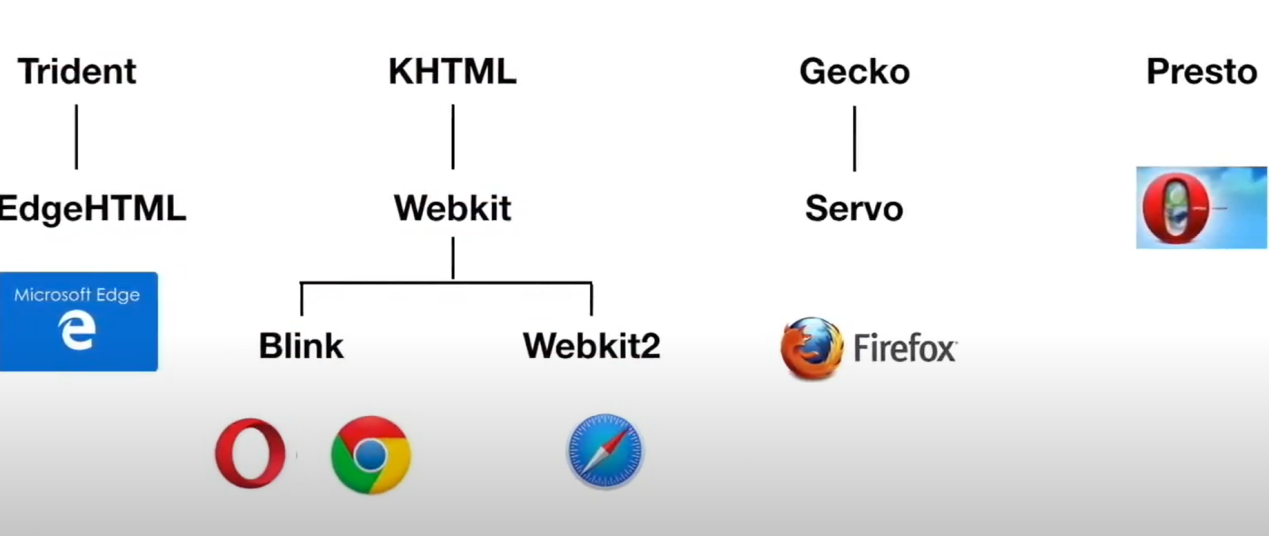
JavaScript引擎V8

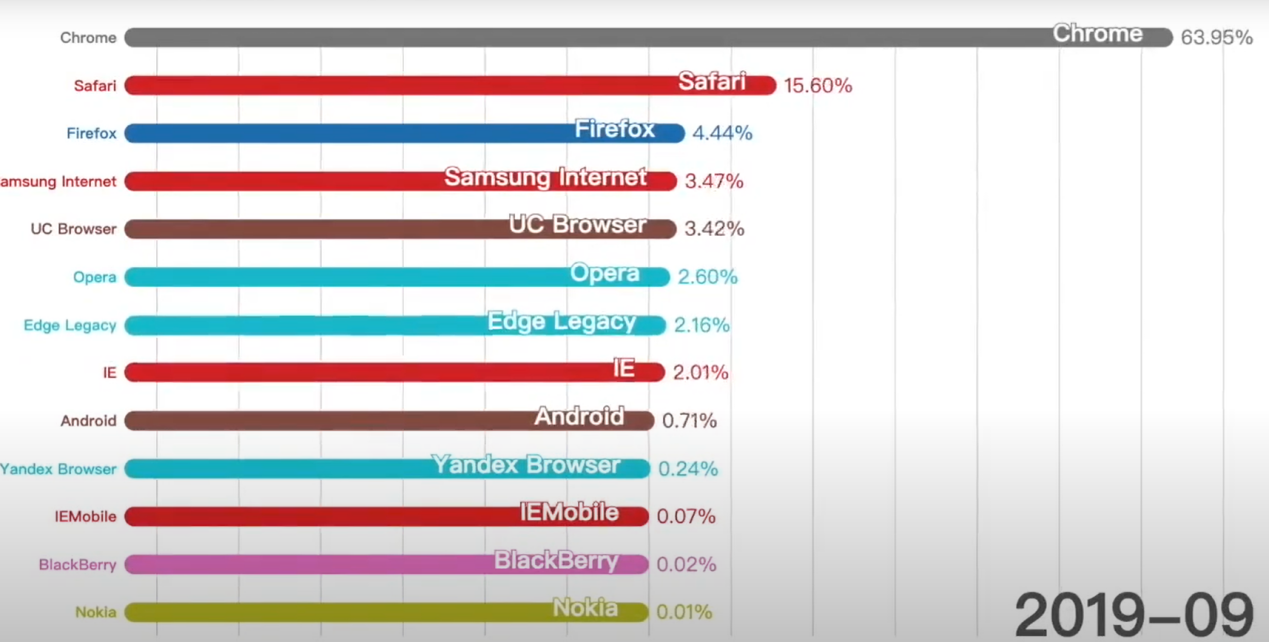
WebKit2

2010年推出了苹果推出，与Chrome理念冲突，2013年Google从Webkit分支分离出来取名为Blink

Gecko(firefox)

2004年 firefox1.0





# **Install**

Install Playwright TestRunner

Install Playwright Browsers

npx playwright install // download new browsers

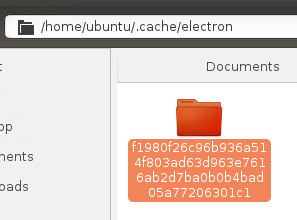
Run all test cases

npx playwright test

## 3.1离线安装前准备

Electron 版本：16.2.0

存放缓存路径: ~/.cache/electron/



## 3.2安装

1：install playwright/test

PLAYWRIGHT\_SKIP\_BROWSER\_DOWNLOAD=1 npm i -D @playwright/test

2：install playwright

PLAYWRIGHT\_SKIP\_BROWSER\_DOWNLOAD=1 npm i -D playwright

如果不下载浏览器，1和2的效果是一样的

npm install packagename --save 或 -S

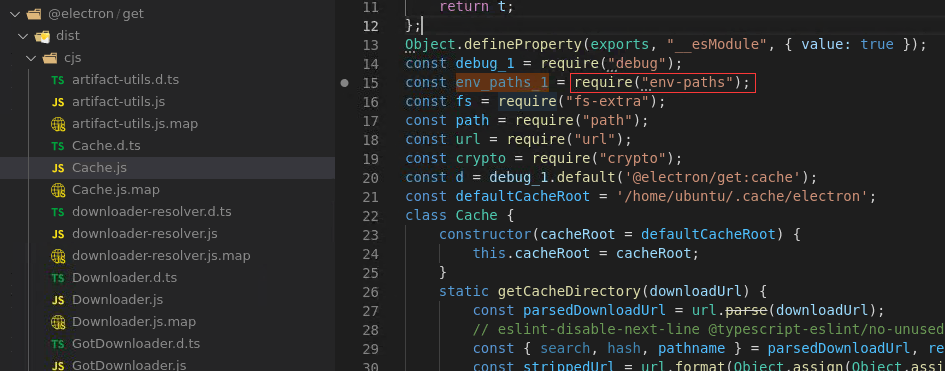
模块的版本信息保存到dependencies（生产环境依赖）中，即你的package.json文件的dependencies字段中；

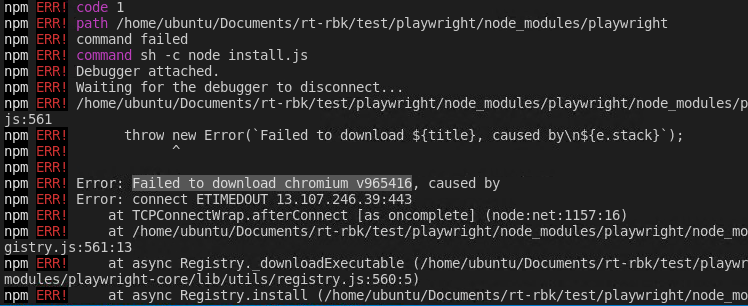
npm install packagename --save-dev 或 -D

模块版本信息保存到devDependencies（开发环境依赖）中，即你的package.json文件的devDependencies字段中

## 3.3踩坑记

* 必须要跳过browser下载
* Electron也需要离线下载，放在缓存目录下面







# **Fixture**

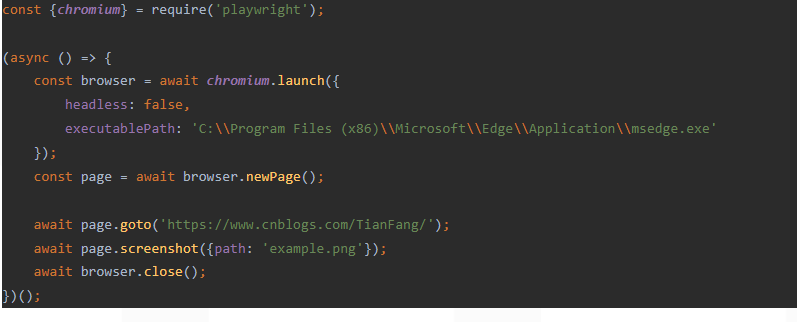
**Build-in**

## Browser

A Browser is created via browserType.launch([options]). An example of using a Browser to create a Page:

*const* { firefox } = require('playwright'); // Or 'chromium' or 'webkit'.  
  
(*async* () => {  
*const* browser = *await* firefox.launch();  
*const* page = *await* browser.newPage();  
*await* page.goto('https://example.com');  
*await* browser.close();  
})();

### 使用已有的浏览器



### 管理浏览器

**Playwright 将 Chromium、WebKit 和 Firefox 浏览器下载到特定于操作系统的缓存文件夹中：**

**%USERPROFILE%\AppData\Local\ms-playwright在 Windows 上**

**~/Library/Caches/ms-playwright在 MacOS 上**

**~/.cache/ms-playwright在 Linux 上**

## Context

BrowserContexts provide a way to operate multiple independent browser sessions.

## Page

## **BrowserName**

  var \_context = await browser.newContext();

  var \_page = await \_context.newPage();

  await \_page.goto('https://example.com');

# Playwright test

### beforeAll

执行一次，中途出错，还会再执行

### beforeEach

每个测试都会执行一次

# Assertions

// Expect a title "to contain" a substring.  
*await* expect(page).toHaveTitle(/Playwright/);

// Expect an attribute "to be strictly equal" to the value.  
*await* expect(page.locator('text=Get Started').first()).toHaveAttribute('href', '/docs/intro');

# Using test hooks

beforeEach

beforeAll

# Persistent authentication

*const* { chromium } = require('playwright');  
  
*const* userDataDir = '/path/to/directory';  
*const* context = *await* chromium.launchPersistentContext(userDataDir, { headless: false });  
// Execute login steps manually in the browser window

**2: Electron**

*PLAYWRIGHT\_SKIP\_BROWSER\_DOWNLOAD*=1 npm i -D playwright

Supported Electron versions are:

* v12.2.0+
* v13.4.0+
* v14+
* ..

**3: Node**

Support Node version: >=12

4: Angular

**4: Current version**

Playwright: v1.19

Node: v16.14.0

Electron: v4.0

Angular: v4.0

1: install playwright

# Run from your project's root directory

npm init playwright

# Or create a new project

npm init playwright new-project

npm i -D @playwright/test

# install supported browsers

npx playwright install

2: install node

cd /usr/local

wget https://nodejs.org/dist/v12.19.0/node-v12.19.0-linux-x64.tar.xz

xz -d node-v12.19.0-linux-x64.tar.xz

tar -xf node-v12.19.0-linux-x64.tar

ln -s /usr/local/node-12.19.0/bin/node /usr/local/bin/node

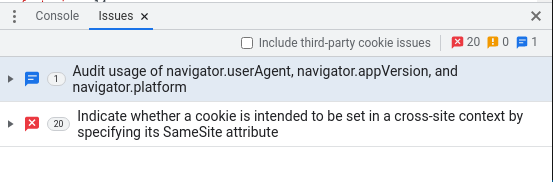
ln -s /usr/local/node-12.19.0/bin/npm /usr/local/bin/npm

/usr/bin

3 install i pm2 –g

**Question**

Q1: Indicate whether a cookie is intended to be set in a cross-site context by specifying its SameSite attribute

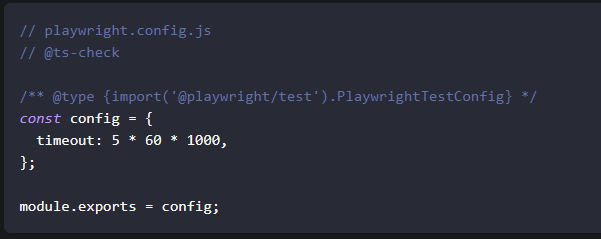


# Command

npx playwright test --timeout 5000

npx playwright test tests/hooks.spec.ts --headed

1. Set test timeout in the config



1. Set timeout for a single test

test.setTimeout(120000);

1. Selectors
2. Text selector

await appWindow.locator('text= English').click();

1. CSS selector

await appWindow.locator('.bet-type-box>div:nth-child(1)').click();

1. Generate tests

npx playwright codegen baidu.com

1. npx playwright test --headed

Use headed browser chromium

1. npx playwright test --headed
2. Run a single test file
3. Run a set of test files
   1. npx playwright test tests/hooks.spec.ts tests/example.spec.ts --headed
4. Run the test with the title

npx playwright test -g "sub" --headed

1. Run tests in headed browser

npx playwright test --headed

1. Disable parallelization

npx playwright test --headed --workers=1

1. Show reporter

npx playwright test --reporter=html

npx playwright show-report

*const* config: PlaywrightTestConfig = {  
reporter: [ ['html', { outputFolder: 'my-report' }] ],  
};

1. Ask for help

npx playwright test --help

# 录制脚本

npx playwright codegen baidu.com

# 参考网站

* <https://www.cnblogs.com/fnng/p/14274960.html?ivk_sa=1024320u>