# Events

## Introduction

Playwright allows listening to various types of events happening on the web page, such as network requests, creation of child pages, dedicated workers etc. There are several ways to subscribe to such events, such as waiting for events or adding or removing event listeners.

## Waiting for event

Most of the time, scripts will need to wait for a particular event to happen. Below are some of the typical event awaiting patterns.

Wait for a request with the specified url using page.waitForRequest():

```
// Start waiting for request before goto. Note no await.
const requestPromise = page.waitForRequest('**/*logo*.png');
await page.goto('https://wikipedia.org');
const request = await requestPromise;
console.log(request.url());
```

Wait for popup window:

```
// Start waiting for popup before clicking. Note no await.
const popupPromise = page.waitForEvent('popup');
await page.getByText('open the popup').click();
const popup = await popupPromise;
await popup.goto('https://wikipedia.org');
```

## Adding/removing event listener

Sometimes, events happen in random time and instead of waiting for them, they need to be handled. Playwright supports traditional language mechanisms for subscribing and unsubscribing from the events:

```
page.on('request', request => console.log(`Request sent:
${request.url()}`));
const listener = request => console.log(`Request finished:
${request.url()}`);
page.on('requestfinished', listener);
await page.goto('https://wikipedia.org');
```

```
page.off('requestfinished', listener);
await page.goto('https://www.openstreetmap.org/');
```

## Adding one-off listeners

If a certain event needs to be handled once, there is a convenience API for that:

```
page.once('dialog', dialog => dialog.accept('2021'));
await page.evaluate("prompt('Enter a number:')");
```

Previous
Evaluating JavaScript

Next
Extensibility