# Comment gérer localStorage dans les jest tests?

144
a

Je continue à obtenir "localStorage n'est pas défini" dans les tests Jest, ce qui a du sens, mais sont mes options? Frapper les murs de briques.

jestjs

f

## Réponses:

141
a

Excellente solution de @chiedo

Cependant, nous utilisons la syntaxe ES2015 et j'ai trouvé que c'était un peu plus propre de l'écrire de cette façon.

f

```
class LocalStorageMock {
  constructor() {
    this.store = {};
  }

  clear() {
    this.store = {};
  }

  getItem(key) {
    return this.store[key] || null;
  }

  setItem(key, value) {
    this.store[key] = value.toString();
  }

  removeItem(key) {
    delete this.store[key];
  }
};

global.localStorage = new LocalStorageMock;
```

— Nickcan

e source

8
f

Devrait probablement faire `value + ''` dans le setter pour gérer correctement les valeurs nulles et non définies

— menehune23

f

Je pense que cette dernière plaisanterie consistait simplement à utiliser cela, `|| null` c'est pourquoi mon test a échoué, car dans mon test, j'utilisais `not.toBeDefined()` . Solution @Chiedo le faire fonctionner à nouveau

— jcubic

Je pense que c'est techniquement un stub :) voir ici pour la version

### Questions similaires:

Comment tester un seul fichier à l'aide de Jest? a 361

Comment exécuter un seul test à l'aide de Jest? a 356

Quelle est la différence entre 'it' et 'test' en plaisantant? a 282

Comment puis-je simuler une importation de module ES6 à l'aide de Jest? a 281

Comment utiliser ESLint avec Jest a 254

f — simulée

100
a

f

Je l'ai compris avec l'aide de ceci: https://groups.google.com/forum/#!topic/jestjs/9EPhuNWVYTg

Configurez un fichier avec le contenu suivant:

```
var localStorageMock = (function() {
  var store = {};
  return {
    getItem: function(key) {
      return store[key];
    },
    setItem: function(key, value) {
      store[key] = value.toString();
    },
    clear: function() {
      store = {};
    },
    removeItem: function(key) {
      delete store[key];
    }
  };
})();
Object.defineProperty(window, 'localStorage', { value: localStorageMock });
```

Ensuite, vous ajoutez la ligne suivante à votre package.json sous vos configs Jest

```
"setupTestFrameworkScriptFile":"PATH_TO_YOUR_FILE",
```

— Chiedo

e source

6
f

Apparemment, avec l'une des mises à jour, le nom de ce paramètre a changé et maintenant il s'appelle "setupTestFrameworkScriptFile"

— Grzegorz Pawlik

2
f

`"setupFiles": [...]` fonctionne aussi bien. Avec l'option array, permet de séparer les simulacres en fichiers séparés. Ex: `"setupFiles": ["<rootDir>/__mocks__/localStorageMock.js"]`

— Stiggler

3
f

La valeur de retour de `getItem` diffère légèrement de ce qui serait renvoyé par un navigateur si aucune donnée n'est définie sur une clé spécifique. appeler `getItem("foo")` quand il n'est pas défini retournera par exemple `null` dans un navigateur, mais `undefined` par ce simulacre - cela provoquait l'échec d'un de mes tests. La solution simple pour moi était de revenir `store[key] || null` dans la `getItem` fonction

— Ben Broadley

f

cela ne fonctionne pas si vous faites quelque chose comme `localStorage['test'] = '123'; localStorage.getItem('test')`

— rob

3
f

J'obtiens l'erreur suivante - la valeur jest.fn () doit être une fonction simulée ou un espion. Des idées?

— Paul Fitzgerald

55
a

f

Si vous utilisez create-react-app, il existe une solution plus simple et directe expliquée dans la documentation .

Créez `src/setupTests.js` et mettez-y ceci:

```
const localStorageMock = {
  getItem: jest.fn(),
  setItem: jest.fn(),
  clear: jest.fn()
};
global.localStorage = localStorageMock;
```

*Contribution de Tom Mertz dans un commentaire ci-dessous:*

Vous pouvez ensuite tester que les fonctions de votre localStorageMock sont utilisées en faisant quelque chose comme

```
expect(localStorage.getItem).toBeCalledWith('token')
// or
expect(localStorage.getItem.mock.calls.length).toBe(1)
```

à l'intérieur de vos tests si vous vouliez vous assurer qu'il a été appelé. Découvrez https://facebook.github.io/jest/docs/en/mock-functions.html

— c4k

e source

f Salut c4k! Pourriez-vous s'il vous plaît donner un exemple comment vous utiliseriez cela dans vos tests?

— Dimo

f Que voulez-vous dire ? Vous n'avez rien à initialiser dans vos tests, cela se moque automatiquement de ce que `localStorage` vous utilisez dans votre code. (si vous utilisez `create-react-app` et tous les scripts automatiques qu'il fournit naturellement)

— c4k

f Vous pouvez ensuite tester que les fonctions de votre localStorageMock sont utilisées en faisant quelque chose comme `expect(localStorage.getItem).toBeCalledWith('token')` ou à l' `expect(localStorage.getItem.mock.calls.length).toBe(1)` intérieur de vos tests si vous vouliez vous assurer qu'il a été appelé. Découvrez facebook.github.io/jest/docs/en/mock-functions.html

— Tom Mertz

10 f pour cela, j'obtiens une erreur - la valeur jest.fn () doit être une fonction simulée ou un espion. Des idées?

— Paul Fitzgerald

3 f Cela ne causera-t-il pas des problèmes si vous avez plusieurs tests qui utilisent `localStorage` ? Ne voudriez-vous pas réinitialiser les espions après chaque test pour éviter un «débordement» dans d'autres tests?

— Brandon Sturgeon

43 a f Actuellement (octobre 19) localStorage ne peut pas être ridiculisé ou espionné par plaisanterie comme vous le feriez habituellement, et comme indiqué dans la documentation de create-react-app. Cela est dû aux modifications apportées à jsdom. Vous pouvez en savoir plus dans le trackers de problèmes jest et jsdom .

Pour contourner le problème, vous pouvez à la place espionner le prototype:

```
// does not work:
jest.spyOn(localStorage, "setItem");
localStorage.setItem = jest.fn();

// works:
jest.spyOn(window.localStorage.__proto__, 'setItem');
window.localStorage.__proto__.setItem = jest.fn();

// assertions as usual:
expect(localStorage.setItem).toHaveBeenCalled();
```

— Bastian Stein

e source

f En fait, cela fonctionne pour moi juste avec spyOn, pas besoin de remplacer la fonction setItem `jest.spyOn(window.localStorage.__proto__, 'setItem');`

— Yohan Dahmani

f Oui, j'ai énuméré les deux comme alternatives, pas besoin de faire les deux.

— Bastian Stein

f Je voulais dire sans le remplacement du setItem aussi 😉

— Yohan Dahmani

f Je ne pense pas comprendre. Pouvez-vous clarifier s'il vous plaît?

— Bastian Stein

1
f Ah oui. Je disais que vous pouvez utiliser soit la première ligne, soit la deuxième ligne. Ce sont des alternatives qui font la même chose. Quelle que soit votre préférence personnelle :) Désolé pour la confusion.

— Bastian Stein

14
a

f ou vous prenez simplement un faux paquet comme celui-ci:

https://www.npmjs.com/package/jest-localstorage-mock

il gère non seulement la fonctionnalité de stockage, mais vous permet également de tester si le magasin a été réellement appelé.

— Aligertor

e source

13
a

f Une meilleure alternative qui gère les `undefined` valeurs (elle n'a pas `toString()` ) et retourne `null` si la valeur n'existe pas. Testé ceci avec la `react` v15, `redux` et `redux-auth-wrapper`

```
class LocalStorageMock {
  constructor() {
    this.store = {}
  }

  clear() {
    this.store = {}
  }

  getItem(key) {
    return this.store[key] || null
  }

  setItem(key, value) {
    this.store[key] = value
  }

  removeItem(key) {
    delete this.store[key]
  }
}

global.localStorage = new LocalStorageMock
```

— Dmitriy

e source

f Merci à Alexis Tyler pour l'idée d'ajouter `removeItem` : developer.mozilla.org/en-US/docs/Web/API/Storage/removeItem

— Dmitriy

f

Croyez que nul et indéfini doivent aboutir à "null" et "undefined" (chaînes littérales)

— menehune23

6

a

f

Si vous recherchez une maquette et non un talon, voici la solution que j'utilise:

```
export const localStorageMock = {
   getItem: jest.fn().mockImplementation(key => localStorageItems[key]),
   setItem: jest.fn().mockImplementation((key, value) => {
       localStorageItems[key] = value;
   }),
   clear: jest.fn().mockImplementation(() => {
       localStorageItems = {};
   }),
   removeItem: jest.fn().mockImplementation((key) => {
       localStorageItems[key] = undefined;
   }),
};

export let localStorageItems = {}; // eslint-disable-line import/no-mutable-exports
```

J'exporte les éléments de stockage pour une initialisation facile. IE je peux facilement le définir sur un objet

Dans les nouvelles versions de Jest + JSDom, il n'est pas possible de définir cela, mais le stockage local est déjà disponible et vous pouvez l'espionner comme ceci:

```
const setItemSpy = jest.spyOn(Object.getPrototypeOf(window.localStorage), 'setItem');
```

— TigerBear

e source

5

a

f

J'ai trouvé cette solution de github

```
var localStorageMock = (function() {
  var store = {};

  return {
    getItem: function(key) {
        return store[key] || null;
    },
    setItem: function(key, value) {
        store[key] = value.toString();
    },
    clear: function() {
        store = {};
    }
  };
})();

Object.defineProperty(window, 'localStorage', {
 value: localStorageMock
});
```

Vous pouvez insérer ce code dans vos setupTests et cela devrait fonctionner correctement.

Je l'ai testé dans un projet avec typesctipt.

— Carlos Huamani

e source

f

pour moi Object.defineProperty a fait l'affaire. L'affectation directe d'objet n'a pas fonctionné. Merci!

— Vicens Fayos

4

Malheureusement, les solutions que j'ai trouvées ici n'ont pas fonctionné pour moi.

a

Donc, je regardais les problèmes de Jest GitHub et j'ai trouvé ce fil

f

Les solutions les plus appréciées étaient celles-ci:

```
const spy = jest.spyOn(Storage.prototype, 'setItem');

// or

Storage.prototype.getItem = jest.fn(() => 'bla');
```

— Christian Saiki

e source

---

f

Mes tests n'ont `window` ni `Storage` défini ni défini. C'est peut-être l'ancienne version de Jest que j'utilise.

— Antrikshy

---

3

a

Comme @ ck4 l'a suggéré, la documentation a une explication claire de l'utilisation `localStorage` dans jest. Cependant, les fonctions simulées ne parvenaient à exécuter aucune des `localStorage` méthodes.

f

Vous trouverez ci-dessous l'exemple détaillé de mon composant react qui utilise des méthodes abstraites pour écrire et lire des données,

```
//file: storage.js
const key = 'ABC';
export function readFromStore (){
    return JSON.parse(localStorage.getItem(key));
}
export function saveToStore (value) {
    localStorage.setItem(key, JSON.stringify(value));
}

export default { readFromStore, saveToStore };
```

*Erreur:*

```
TypeError: _setupLocalStorage2.default.setItem is not a function
```

**Correction:**
Ajouter ci - dessous pour fonction simulée plaisanterie (chemin: `.jest/mocks/setUpStore.js` )

```
let mockStorage = {};

module.exports = window.localStorage = {
  setItem: (key, val) => Object.assign(mockStorage, {[key]: val}),
  getItem: (key) => mockStorage[key],
  clear: () => mockStorage = {}
};
```

L'extrait est référencé à partir d' ici

— Mad-D

e source

---

3

a

Vous pouvez utiliser cette approche pour éviter de vous moquer.

```
Storage.prototype.getItem = jest.fn(() => expectedPayload);
```

f

— Sanath

e source

---

2

a

Riffé quelques autres réponses ici pour le résoudre pour un projet avec Typescript. J'ai créé un LocalStorageMock comme ceci:

f

```
export class LocalStorageMock {

    private store = {}

    clear() {
        this.store = {}
    }

    getItem(key: string) {
        return this.store[key] || null
    }

    setItem(key: string, value: string) {
        this.store[key] = value
    }

    removeItem(key: string) {
        delete this.store[key]
    }
}
```

Ensuite, j'ai créé une classe LocalStorageWrapper que j'utilise pour tous les accès au stockage local dans l'application au lieu d'accéder directement à la variable de stockage local globale. Facilité de mise en place de la maquette dans l'emballage pour les tests.

— CorayThan

e source

---

2

a

f

```
    describe('getToken', () => {
    const Auth = new AuthService();
    const token =
'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Ik1yIEpvc2VwaCIsImlkIjoiNWQwYjk1Mzg2NTVh
OTQ0ZjA0NjE5ZTA5IiwiZW1haWwiOiJ0cmV2X2pvc0Bob3RtYWlsLmNvbSIsInByb2ZpbGVVc2VybmFtZSI6Ii9tcmpvc2
VwaCIsInByb2ZpbGVJbWFnZSI6Ii9Eb3Nlbi10LUdpci1sb29rLWN1dGUtbnVrZWNhdDMxNnMtMzExNzAwNDYtMTI4MC04
MDAuanBnIiwiaWF0IjoxNTYyMzE4NDA0LCJleHAiOjE1OTM4NzYwMDR9.YwU15SqHMh1nO51eSa0YsOK-
YLlaCx6ijceOKhZfQZc';
    beforeEach(() => {
        global.localStorage = jest.fn().mockImplementation(() => {
            return {
                getItem: jest.fn().mockReturnValue(token)
            }
        });
    });
    it('should get the token from localStorage', () => {

        const result  = Auth.getToken();
        expect(result).toEqual(token);

    });
});
```

Créez une maquette et ajoutez-la à l' `global` objet

— Trevor Joseph

e source

---

2

a

Vous devez simuler le stockage local avec ces extraits

// localStorage.js

f

```
var localStorageMock = (function() {
    var store = {};

    return {
        getItem: function(key) {
            return store[key] || null;
        },
        setItem: function(key, value) {
            store[key] = value.toString();
        },
        clear: function() {
            store = {};
        }
    };

})();

Object.defineProperty(window, 'localStorage', {
     value: localStorageMock
});
```

Et dans la configuration de plaisanterie:

```
"setupFiles":["localStorage.js"]
```

N'hésitez pas à demander quoi que ce soit.

— Codeur mince

e source

---

1

a

La solution suivante est compatible pour les tests avec une configuration TypeScript, ESLint, TSLint et Prettier plus stricte `{ "proseWrap": "always", "semi": false, "singleQuote": true, "trailingComma": "es5" }` :

f

```
class LocalStorageMock {
  public store: {
    [key: string]: string
  }
  constructor() {
    this.store = {}
  }

  public clear() {
    this.store = {}
  }

  public getItem(key: string) {
    return this.store[key] || undefined
  }

  public setItem(key: string, value: string) {
    this.store[key] = value.toString()
  }

  public removeItem(key: string) {
    delete this.store[key]
  }
}
/* tslint:disable-next-line:no-any */
;(global as any).localStorage = new LocalStorageMock()
```

HT / https://stackoverflow.com/a/51583401/101290 pour savoir comment mettre à jour global.localStorage

— Beau Smith

e source

---

1

Pour faire la même chose dans le TypeScript, procédez comme suit:

a

f

Configurez un fichier avec le contenu suivant:

```
let localStorageMock = (function() {
  let store = new Map()
  return {

    getItem(key: string):string {
      return store.get(key);
    },

    setItem: function(key: string, value: string) {
      store.set(key, value);
    },

    clear: function() {
      store = new Map();
    },

    removeItem: function(key: string) {
        store.delete(key)
    }
  };
})();
Object.defineProperty(window, 'localStorage', { value: localStorageMock });
```

Ensuite, vous ajoutez la ligne suivante à votre package.json sous vos configs Jest

```
"setupTestFrameworkScriptFile":"PATH_TO_YOUR_FILE",
```

Ou vous importez ce fichier dans votre scénario de test où vous souhaitez simuler le stockage local.

— vs_lala

e source

---

0

a

f

Cela a fonctionné pour moi,

```
delete global.localStorage;
global.localStorage = {
getItem: () =>
 }
```

— prathique

e source

En utilisant notre site, vous reconnaissez avoir lu et compris notre politique liée aux cookies et notre politique de confidentialité.

Licensed under cc by-sa 3.0 with attribution required.