Meteor Installation Guide

1. Meteor Project
2. React
3. Material-UI  
   meteor npm install material-ui@next
4. Jest, Enzyme for Test
   1. Some special changes in ordert o get it working with meteor
5. Type-Script  
   [<https://medium.com/@birkskyum/modern-meteor-development-with-typescript-1-integrate-typescript-with-meteor-905c13edbec8>]  
   
   1. Add type script compiler  
      meteor add barbatus:typescript
   2. Installing type definitions

meteor npm install @types/meteor –save

npm install --save @types/material-ui. //not for 1.0.0 beta version next

* 1. Configure Typescript   
     Create tsconfig.json

|  |
| --- |
| { |
|  | "compilerOptions": { |
|  | "target": "es5", |
|  | "module": "commonjs", |
|  | "moduleResolution": "node", |
|  | "alwaysStrict": true, |
|  | "sourceMap": true, |
|  | "jsx": "react", |
|  | "allowJs": false, |
|  | "noImplicitAny": false, |
|  | "strictNullChecks": false |
|  | }, |
|  | "include": [ |
|  | "client/\*\*/\*", |
|  | "server/\*\*/\*" |
|  | ], |
|  | "exclude": [ |
|  | "node\_modules", |
|  | ".meteor" |
|  | ] |
|  | } |

* 1. npm install -g tslint typescript
  2. Install TSLInt for visual code
  3. Configre TSLInter via  
     tslint.json
  4. npm install @types/prop-types
  5. yarn add -D typescript jest ts-jest @types/jest  
     (Quelle: <https://medium.com/@mtiller/debugging-with-typescript-jest-ts-jest-and-visual-studio-code-ef9ca8644132>)
  6. yarn add enzyme-adapter-react-16 enzyme
  7. yarn add -D jest react-test-renderer @types/enzyme
  8. yarn add -D @types/enzyme-adapter-react-16
  9. Configuring Jest + Enzyme + Typescript  
     (Quelle: <https://medium.com/@mateuszsokola/configuring-react-16-jest-enzyme-typescript-7122e1a1e6e8>)
     1. create three files:
        1. **test-preprocessor.js** — to transpile test files written in Typescript to JavaScript.
        2. **test-setup.js** — to setup enzyme to use React 16.
        3. **test-shim.js** — to get rid of warnings regarding missing browser polyfills.
     2. modify the package.json file and add the jest property to its content.

|  |
| --- |
| "name":"open joblist", |
|  | ... |
|  | "jest": { |
|  | "setupFiles": [ |
|  | "<rootDir>/test-shim.js", |
|  | "<rootDir>/test-setup.js" |
|  | ], |
|  | "moduleFileExtensions": [ |
|  | "ts", |
|  | "tsx", |
|  | "js" |
|  | ], |
|  | "transform": { |
|  | "^.+\\.(ts|tsx)$": "<rootDir>/test-preprocessor.js" |
|  | }, |
|  | "testMatch": [ |
|  | "\*\*/\_\_tests\_\_/\*.(ts|tsx|js)" |
|  | ] |
|  | }, |

* 1. Sdf

Quellen:

# React Design Roles

<https://www.typescriptlang.org/docs/handbook/declaration-files/do-s-and-don-ts.html>

**All React components must act like pure functions with respect to their props.**

State allows React components to change their output over time in response to user actions, network responses, and anything else, without violating this rule.

While this.props is set up by React itself and this.state has a special meaning, you are free to add additional fields to the class manually if you need to store something that is not used for the visual output.

If you don’t use something in render(), it shouldn’t be in the state.

### Do Not Modify State Directly

For example, this will not re-render a component:

// Wrong

this.state.comment = 'Hello';

Instead, use setState():

// Correct

this.setState({comment: 'Hello'});

The only place where you can assign this.state is the constructor.

[props are read-only](https://reactjs.org/docs/components-and-props.html#props-are-read-only)

## React Component

I always create a props Interface for each component, even if it's blank. It keeps things consistant and allows me to easily add props later if needed.

Interface IProps { }

class foo extends React.Component<IProps, any> {

}

### with state

class Foo extends React.Component<IProps, IState> {}

|  |
| --- |
|  |
|  |

### without state

class FirstComponent extends React.Component <{}> {

|  |
| --- |
| class UserComponent extends React.Component<UserInterface, {}> { |

### With generic form [Quelle: <https://blog.mojotech.com/typescript-generic-react-components/>]

class Form<FormFields> extends React.Component<Props<FormFields>, State<FormFields>> {

### ???