BINAR ACADEMY

State Management dengan Redux

Gold - Chapter 4 - Topic 1

Selamat datang di Chapter 4 Topic 1 online course React Native dari Binar Academy!

Haloo, guys! ✌

Kayaknya ada yang sudah tambah familiar dengan React Native nih. Mantap! Biar makin jago, di Chapter 4 ini kita lanjut belajar React Native lagi yuk~

Di chapter sebelumnya kamu sudah belajar tentang RESTFul API, maka di chapter ini kamu akan belajar tentang Redux. Apa itu Redux?

Cari tahu di materi ini!

Detailnya, kita bakal bahas hal-hal berikut ini:

- Pengenalan Redux
- Cara aplikasi Redux
- Komponen Redux

Guys, masih ingat nggak sama state?

Itu lhoo.. yang bikin aplikasimu bisa jadi lebih dinamis dan interaktif.

State dipakai misalnya saat kamu mau melakukan perubahan data yang akan ditampilkan pada UI aplikasimu.

Walaupun state bikin aplikasi jadi asyik diakses, tapi ada satu kelemahannya nih, yaitu state hanya akan hidup di dalam komponen itu sendiri.

Eh gimana? Apakah state anak introvert? Hehe canda introvert.

Oke, serius, bagaimana maksudnya ya?

Jadi kayak gini contohnya..

Ketika kamu punya sebuah state isLogin pada komponen <Home /> maka state isLogin hanya akan tersedia di komponen itu saja dan nggak bisa dipakai oleh komponen lainnya.

```
import { useState } from 'react'

const Home = () => {

  const [isLogin, setIsLogin] = useState(false)

  return (

  <View>
    <Text>
        {isLogin}
    </Text>
   </View>

  )
}
```

Nah untuk mengatasi masalah tersebut, kamu memerlukan bantuan library bernama **Redux**.

Ini dia Redux, yuk kenalan~

Redux merupakan sebuah state management yang dapat membuat sebuah state pada aplikasi **menjadi global** atau dengan kata lain **bisa diakses oleh komponen lainnya.**

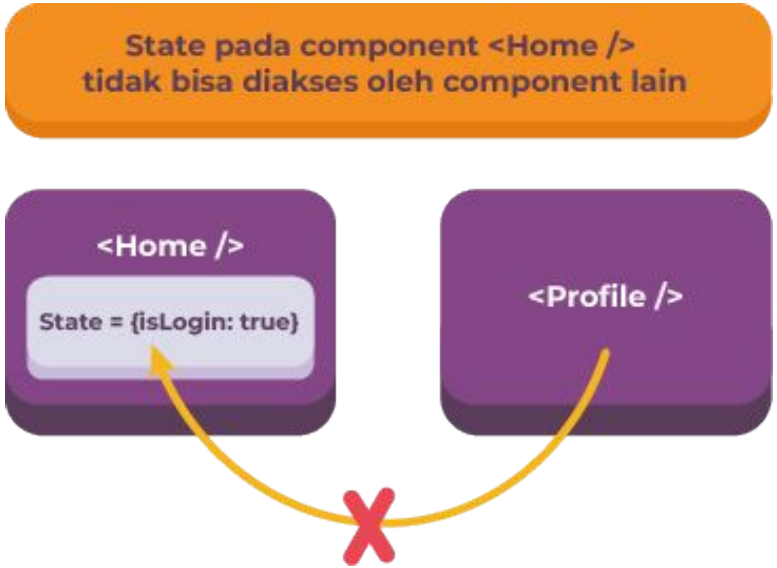
Cakepp..

Redux

Oke kita balik dulu ya sedikit..

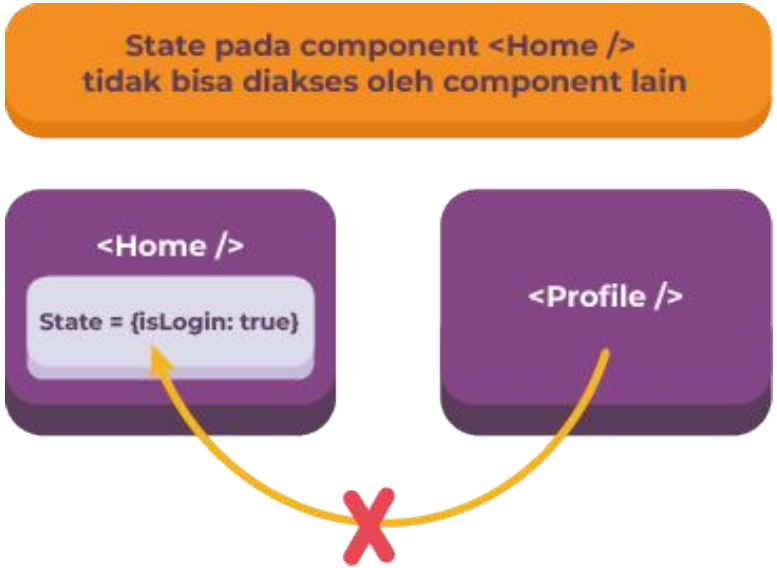
Seperti yang sudah disebutkan sebelumnya, ada satu kelemahan pada state yaitu **suatu state hanya akan hidup di dalam komponen itu sendiri.**

Contoh ketika kamu mempunyai sebuah state **isLogin** pada komponen **<Home />** maka state **isLogin** hanya akan tersedia di komponen itu saja dan nggak bisa dipakai oleh komponen lainnya.



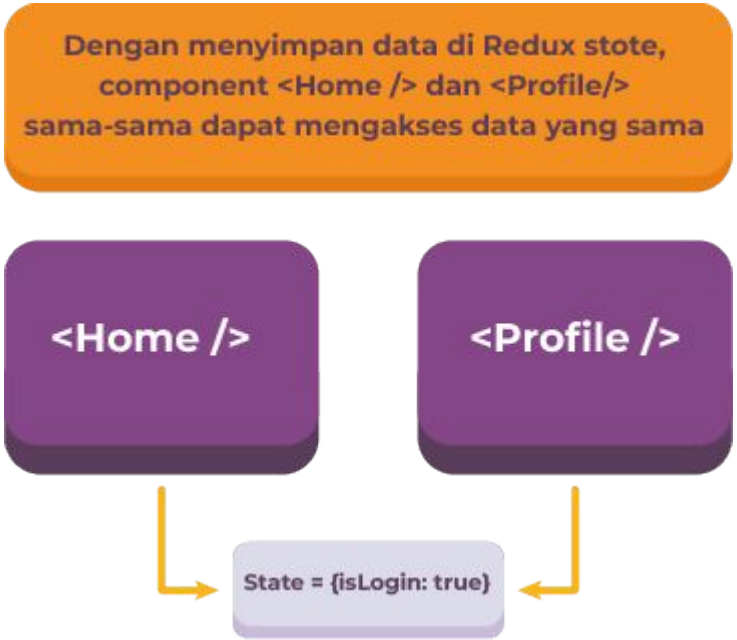
Jadi kalau component <Profile /> mau mengakses data di dalam state isLogin tapi isLogin ini ada di <Home/>, maka hal itu tidak bisa dilakukan.

Karena balik lagi ke peraturan awal state, state cuma bisa diakses oleh komponen yang membuat state tersebut.



Kalau kamu mau bikin **state isLogin dan bikin state tersebut bisa diakses oleh kompen <Profile />** maka hal tersebut bisa kamu lakukan dengan **Redux Store.**

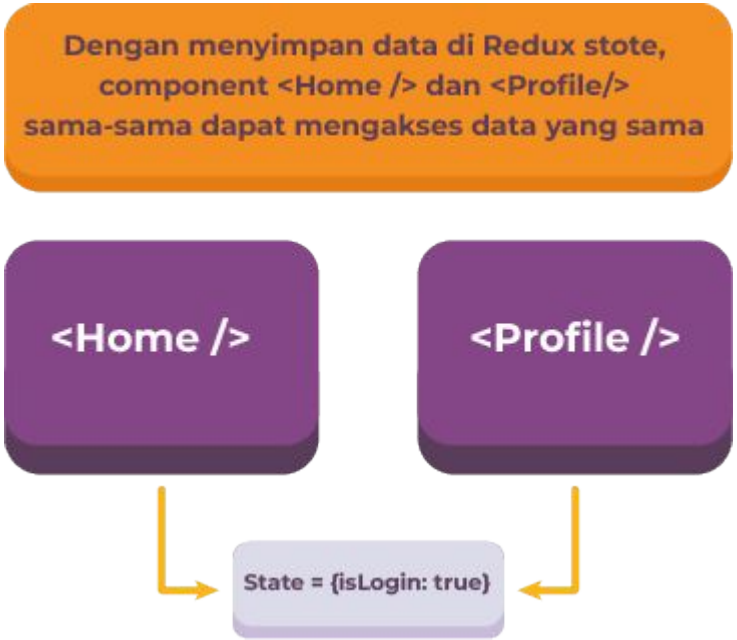
Bagaimana tuh?



Begini..

Caranya adalah dengan menyimpan data yang mau kamu bagikan ke component lain pada redux store, sehingga saat terjadi perubahan pada state **isLogin**, baik komponen **<Profile />** maupun **<Home />** akan mendapatkan data yang sama.

Singkatnya, **kamu membuat satu database khusus** yang nantinya bisa di akses oleh semua komponen.



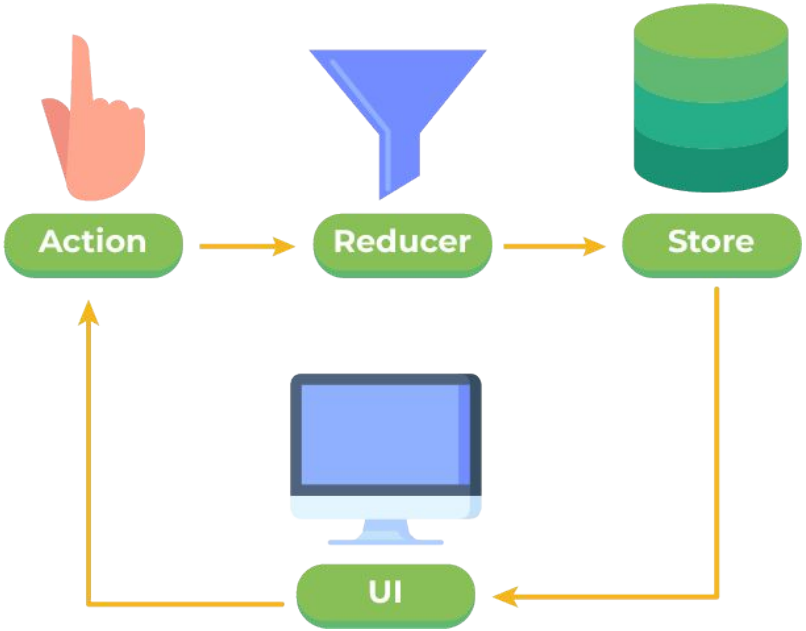
Redux terdiri dari beberapa komponen, guys! Apa saja?

Lanjut kita bahas~

Tiga komponen utama Redux:

- Action
- Reducer
- Store

Kita bahas satu persatu yuks!



Action

Action pada dasarnya hanyalah sebuah object yang memiliki property dengan nama **type.**

Action-lah yang akan menjadi penghubungmu untuk berkomunikasi dengan **Reducer**.

```
const iniContohAction = {
  type: 'INI_ACTION_TYPE',
  data: {nama: "Ikan Hiu makan bakwan"}
}
```

Jika diperlukan, sebuah Action juga bisa memiliki properti kedua. Properti kedua ini biasanya digunakan untuk menyimpan data/payload yang kamu perlukan.

Kesimpulannya, **Action adalah sebuah object yang memiliki dua properti**, yaitu type dan data.

```
const iniContohAction = {
  type: 'INI_ACTION_TYPE',
  data: {nama: "Ikan Hiu makan bakwan"}
}
```

Pengertian Action

Action adalah akibat yang disebabkan oleh sebuah interkasi.

Sebagai contoh, ketika user menekan tombol Register pada UI, maka React Native akan mentrigger atau memicu sebuah Action dengan type REGISTER_USER.

Nah, ketika Action dipicu, selanjutnya Redux akan menjalankan kode-kode yang ada di reducer.

Jadi intinya, Action adalah pemicu proses pada Redux.

Agar lebih dinamis dan memudahkanmu dalam membacanya, biasanya sebuah Action ditulis ke dalam sebuah bentuk function seperti ini:

```
const ADD_DATA = (newData) => {
  return {
    type: '@ADD_DATA',
    data: newData
  }
}

export {
    ADD_DATA
}
```

BINAR

Pakai analogi deh supaya gampang dipahami..

Seperti sebuah restoran, Action bisa kamu umpamakan sebagai seorang konsumen yang akan order menu pesanan.

```
const ADD_DATA = (newData) => {
  return {
    type: '@ADD_DATA',
    data: newData
  }
}

export {
    ADD_DATA
}
```

Konsumen akan menentukan jenis makanan yang ingin ia pesan (Type) dan konsumen bisa menentukan apa yang ingin diubah dengan menentukan data/Payload dari pesanan itu sendiri.

Setelah customer selesai menentukan pesanan maka waiter akan mengirimkan pesanan tersebut kepada Chef.

Nah, Dispatch ini lah yang akan bertugas untuk meneruskan Action kepada Reducer.

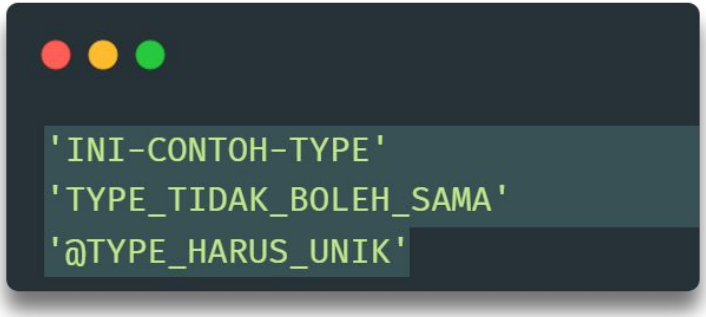
```
const ADD_DATA = (newData) => {
  return {
    type: '@ADD_DATA',
    data: newData
  }
}

export {
    ADD_DATA
}
```

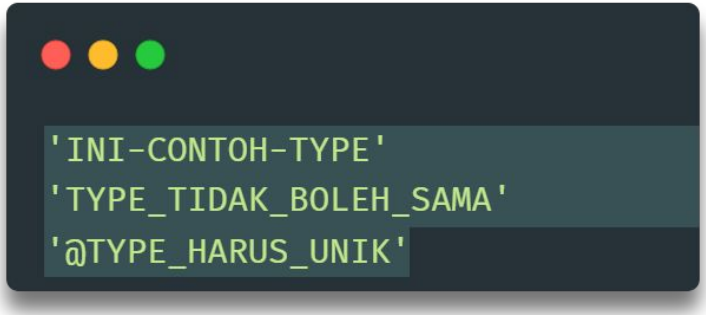
Type

Pada dasarnya Type merupakan sebuah ID atau pembeda antara sebuah **Action** dengan **Action** lainnya.

Type akan selalu dikirimkan oleh Action ketika dilakukannya Dispatch.



Fungsi Type adalah **agar Reducer bisa bedain State mana yang harus diubah berdasarkan Type yang dibawa oleh Action tersebut**, sehingga Reducer hanya akan mengubah State yang sudah ditentukan berdasarkan dengan Type-nya.



Type sangatlah sederhana karena Type nggak lain dan nggak bukan hanyalah sebuah string.

Kalau kamu lihat pada contoh sebelumnya, kamu bisa mengartikan bahwa State **'message'** hanya akan berubah ketika terdapat sebuah **Action** dengan **Type 'ADD_DATA'**

Penulisan type umumnya selalu menggunakan huruf kapital semua, karena berfungsi untuk membedakan antara type dengan deklarasi variabel dengan strings.

```
const initialState = {
  message: '',
};
const reducer = (prevState = initialState, action) => {
  if (action.type === 'ADD_DATA') {
    return { ...prevState, data };
  }
  return prevState;
};

export default reducer;
```

Store

Fungsi Store pada dasarnya adalah sebagai **penghubung atau bagian yang bertugas dalam menggabungkan antara Action dengan Reducer** untuk akhirnya dapat bekerja sama sebagai State Management.

```
const initialState = {
  message: ''
}
const reducer = (prevState = initialState, action) => {
  if (action.type === '@ADD_DATA') {
    return { ...prevState, action.data }
  }
  return prevState
}


export default reducer
```

3 fungsi utama store adalah:

1. Menyimpan keseluruhan State

2. Mengakses State

3. Menjalankan Reducer untuk melakukan perubahan State

File Store ini biasanya ada di level paling atas pada aplikasi kita karena yang akan bertugas sebagai induk utama dari Redux.

```
const initialState = {
  message: ''
}
const reducer = (prevState = initialState, action) => {
  if (action.type === '@ADD_DATA') {
    return { ...prevState, action.data }
  }
  return prevState
}


export default reducer
```

Reducer

Reducer merupakan sebuah bagian dari Redux yang berperan dalam **mengubah State menjadi respon** ketika terdapat sebuah Action yang di-dispatch.

Pada dasarnya perubahan State di Redux terjadi di dalam Reducer. Reducer akan berjalan ketika ada Action yang di dispatch dan melakukan perubahan data sesuai dengan Action-nya.

```
const initialState = {
  message: ''
}
const reducer = (prevState = initialState, action) => {
  if (action.type === '@ADD_DATA') {
    return { ...prevState, action.data }
  }
  return prevState
}


export default reducer
```

Jadi guys, singkatnya, **Reducer adalah sebuah function yang akan mengubah State lama dan mengembalikan State baru.**

```
const initialState = {
  message: ''
}
const reducer = (prevState = initialState, action) => {
  if (action.type === '@ADD_DATA') {
    return { ...prevState, action.data }
  }
  return prevState
}


export default reducer
```

Kalau sebelumnya kamu sudah membayangkan **Action sebagai seorang customer** yang akan menentukan pesanan dan **Dispatch adalah seorang waiter** yang akan meneruskan pesanan customer kepada Chef, maka **Reducer adalah Chef** yang bertugas dan bertanggung jawab untuk menerima pesanan dan membuat apa yang diinginkan oleh customer/Action.

Setelah pesanan tersebut dibuat, maka seorang chef/Reducer akan mengembalikan atau memberikan data berupa State baru yang telah selesai diubah.

```
const initialState = {
  message: ''
}
const reducer = (prevState = initialState, action) => {
  if (action.type === '@ADD_DATA') {
    return { ...prevState, action.data }
  }
  return prevState
}


export default reducer
```

Redux - Reducer

Yang terjadi kalau sebuah Action dijalankan atau di dispatch() adalah function akan berjalan.

Selanjutnya pada Reducer akan dilakukan pengecekan apakah type dari Action-nya adalah @ADD_DATA, maka Reducer akan melakukan perubahan tersebut dan mengembalikan nilai State yang baru.

```
const initialState = {
  message: ''
}
const reducer = (prevState = initialState, action) => {
  if (action.type === '@ADD_DATA') {
    return { ...prevState, action.data }
  }
  return prevState
}


export default reducer
```

Saatnya kita

Quiz!

BINAR

1. Apa yang terjadi apabila aplikasi kamu tidak pake redux?

A. Komponen-komponen lain tidak bisa mengakses data pada suatu state yang terdapat pada suatu komponen

B. Komponen tidak efektif kaena bisa dapat mengakses multiple data

C. Komponen-komponen lain bisa mengakses data pada suatu state, tertapi akan menyebabkan “tumburan” Ketika mengakses datanya bersamaan.

BINAR

1. Apa yang terjadi apabila aplikasi kamu tidak pake redux?

A. **Komponen-komponen lain tidak bisa mengakses data pada suatu state yang terdapat pada suatu komponen**

B. Komponen tidak efektif kaena bisa dapat mengakses multiple data

C. Komponen-komponen lain bisa mengakses data pada suatu state, tertapi akan menyebabkan “tumburan” Ketika mengakses datanya bersamaan.

Jika tidak menggunakan redux, kamu tidak bisa bertukar data atau mengakses data pada komponen lain

BINAR

2. Mana syntax kode dibawah ini yang merupakan ‘action’ pada Redux?

A.

```
export const getDataGempaTerbaru = () =>{
  return {
    type: "FETCH_DATA_GEMBA_TERBARU"
  }
}
```

B.

```
"FETCH_DATA_GEMBA_TERBARU"
```

C.

```
const initialState = {
  listDataGempa: []
};
const reducer = (prevState = initialState, action) => {
  if (action.type === 'FETCH_DATA_GEMBA_TERBARU') {
    return { ...prevState: listDataGempa: null};
  }
  return prevState;
};

export default reducer;
```

BINAR

2. Mana syntax kode dibawah ini yang merupakan ‘action’ pada Redux?

A.

```
export const getDataGempaTerbaru = () =>{
  return {
    type: "FETCH_DATA_GEMBA_TERBARU"
  }
}
```

B.

```
"FETCH_DATA_GEMBA_TERBARU"
```

C.

```
const initialState = {
  listDataGempa: []
};
const reducer = (prevState = initialState, action) => {
  if (action.type === 'FETCH_DATA_GEMBA_TERBARU') {
    return { ...prevState, listDataGempa: null};
  }
  return prevState;
};

export default reducer;
```

Action adalah sebuah function yang mereturn object yang pada object tersebut terdapat properti type

BINAR

3. Manakah pernyataan-pernyataan yang benar di bawah ini?

A. State pada suatu komponen dapat dibaca oleh komponen lain, tapi tidak bisa di update

B. State pada suatu komponen dapat di update oleh komponen lain

C. State pada suatu komponen tidak dapat dibaca dan diupdate oleh komponen lain

BINAR

3. Manakah pernyataan-pernyataan yang benar di bawah ini?

A. State pada suatu komponen dapat dibaca oleh komponen lain, tapi tidak bisa di update

B. State pada suatu komponen dapat di update oleh komponen lain

C. State pada suatu komponen tidak dapat dibaca dan diupdate oleh komponen lain

State pada suatu komponent tidak dapat dibaca dan diupdate oleh komponen lain

4. Apabila kamu memiliki case, kamu punya komponen dengan nama <Profile /> dan punya komponen <About />, nah di komponen <Profile /> ada state dengan nama fullName yang berisikan infromasi nama lengkap user. Bagaimana caranya agar nama lengkap user yang terdapat pada state fullName ini dapat ditampilkan didalam komponen <About /> ?

A. Menggunakan props pada komponen <About />

B. Memindahkan state fullName ke dalam redux-store.

C. Membuat state yang sama (fullName) pada komponen <About />

4. Apabila kamu memiliki case, kamu punya komponen dengan nama <Profile /> dan punya komponen <About />, nah di komponen <Profile /> ada state dengan nama fullName yang berisikan infromasi nama lengkap user. Bagaimana caranya agar nama lengkap user yang terdapat pada state fullName ini dapat ditampilkan didalam komponen <About /> ?

A. Menggunakan props pada komponen <About />

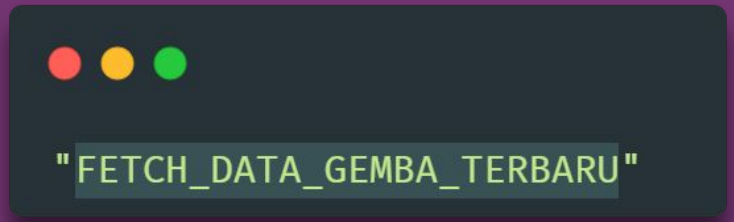
B. Memindahkan state fullName ke dalam redux-store.

C. Membuat state yang sama (fullName) pada komponen <About />

Dengan memindahkan state fullName ke redux-store, maka semua komponen akan dapat mengakses state fullName tersebut

BINAR

5. Syntax pada gambar di bawah merupakan contoh dari?



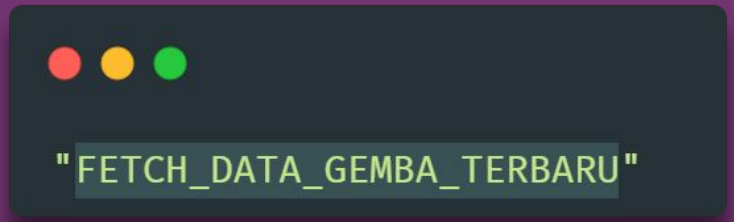
A. Reducer

B. Action

C. Type

BINAR

5. Syntax pada gambar di bawah merupakan contoh dari?



A. Reducer

B. Action

C. **Type**

Type adalah sebuah string dan bisa dibilang unik id yang merncerminkan setiap action pada redux

BINAR

6. Apa perbedaan function getDataGempaTerbaru pada gambar 1 dan gambar 2?

Gambar 1

```
const getDataGempaTerbaru = () => {
  return {
    type: "FETCH_DATA_GEMBA_TERBARU",
    data: null
  }
}
```

Gambar 2

```
const getDataGempaTerbaru = () => ({
    type: "FETCH_DATA_GEMBA_TERBARU",
    data: null
})
```

A. Gambar 2 tidak akan mereturn sebuah object, sedangkan Gambar 1 mereturn sebuah object

B. Gambar 1 tidak akan mereturn sebuah object, sedangkan Gambar 2 mereturn sebuah object

C. Tidak ada perbedaan, Gambar 1 dan 2 sama-sama akan mereturn sebuah object

BINAR

6. Apa perbedaan function getDataGempaTerbaru pada gambar 1 dan gambar 2?

Gambar 1

```
const getDataGempaTerbaru = () => {
  return {
    type: "FETCH_DATA_GEMBA_TERBARU",
    data: null
  }
}
```

Gambar 2

```
const getDataGempaTerbaru = () => ({
    type: "FETCH_DATA_GEMBA_TERBARU",
    data: null
})
```

A. Gambar 2 tidak akan mereturn sebuah object, sedangkan Gambar 1 mereturn sebuah object

B. Gambar 1 tidak akan mereturn sebuah object, sedangkan Gambar 2 mereturn sebuah object

C. Tidak ada perbedaan, Gambar 1 dan 2 sama-sama akan mereturn sebuah object

Kedua function tersebut adalah arrow function, dan sama-sama akan mereturn sebuah object

Terima Kasih!

Next Topic

loading...

BINAR