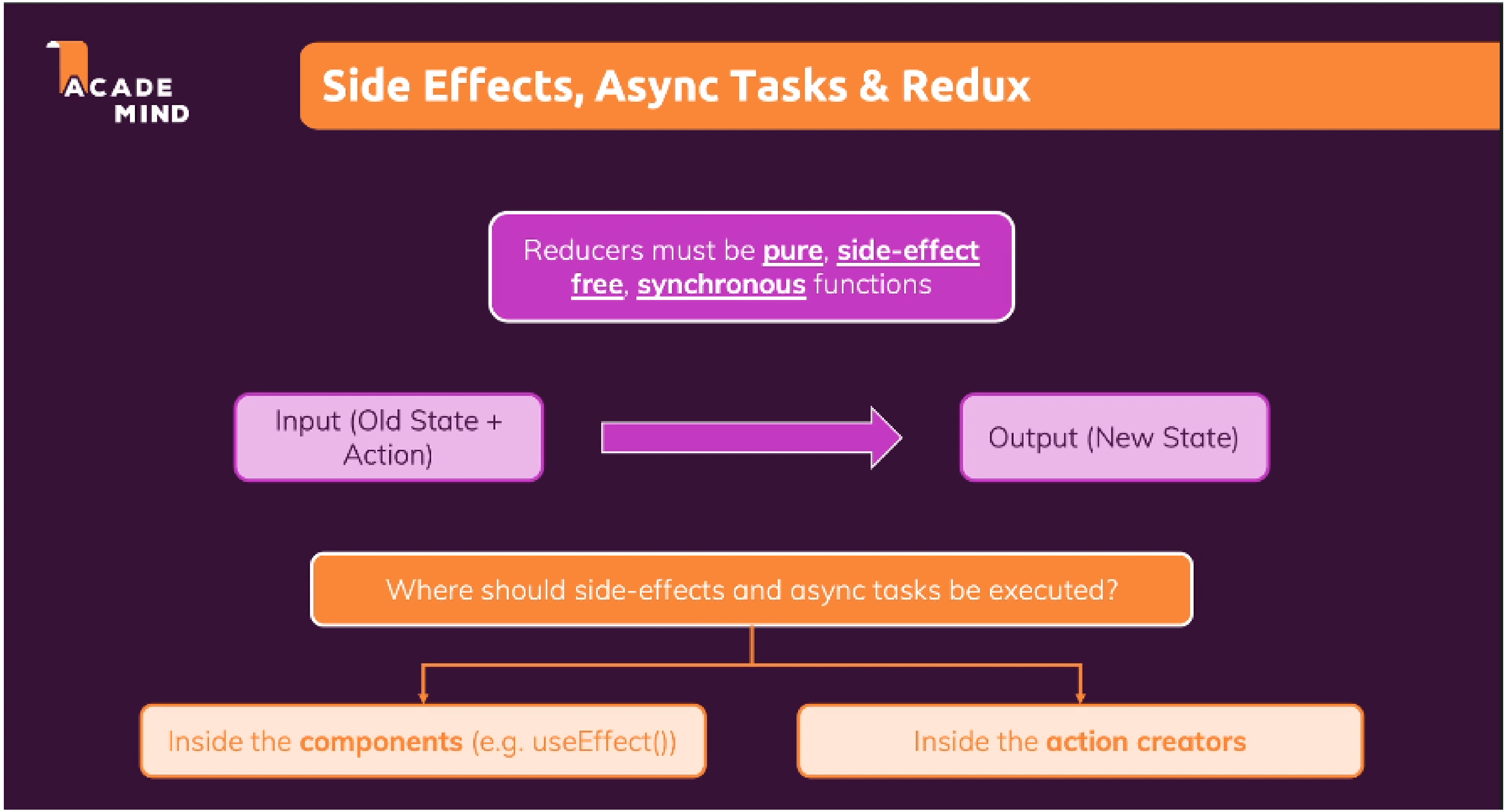


Side Effects, Async Tasks & Redux

Reducers must be **pure**, **side-effect free**, **synchronous** functions

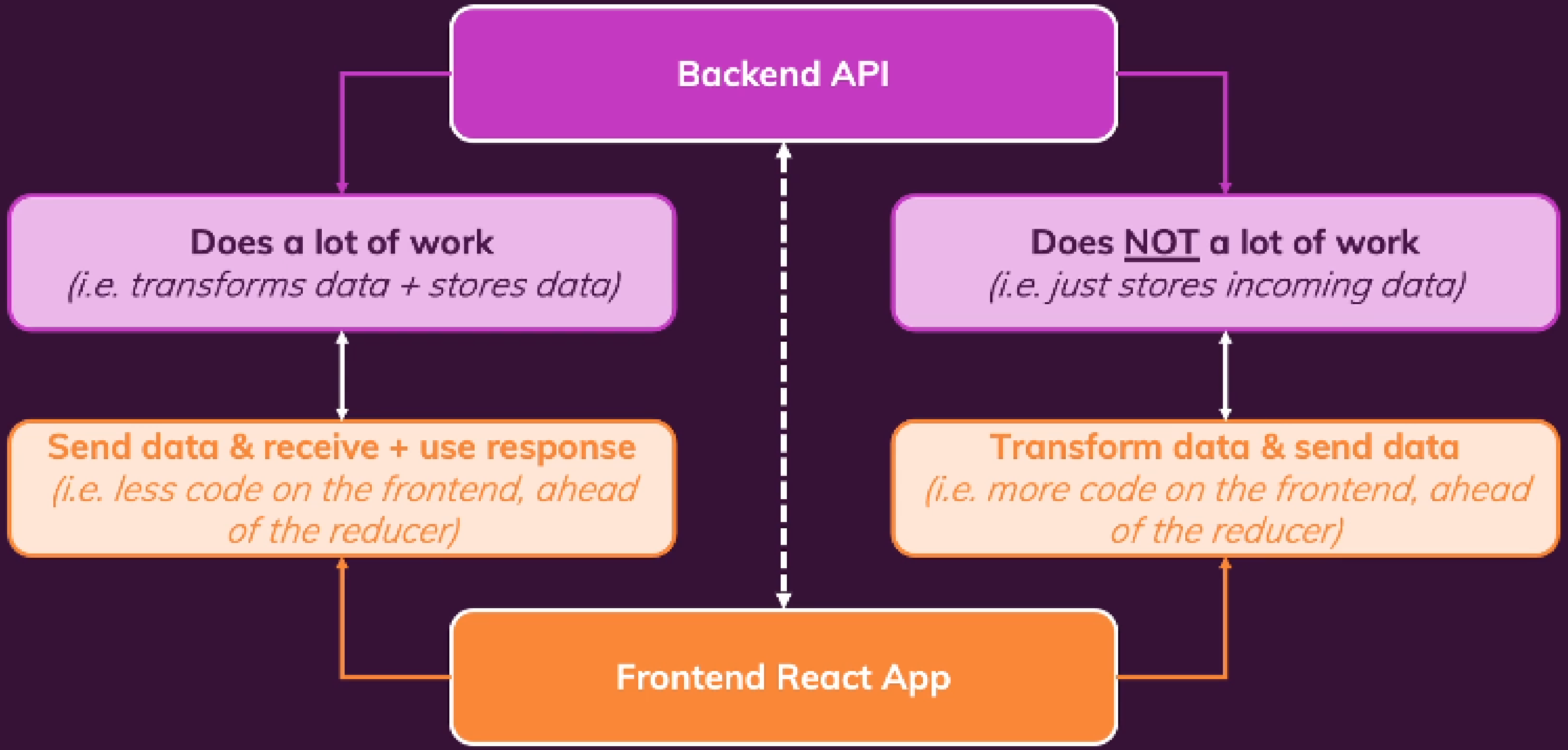
Input (Old State + Action) → Output (New State)

Where should side-effects and async tasks be executed?

Inside the **components** (e.g. useEffect())

Inside the **action creators**

Frontend Code Depends On Backend Code



Backend API

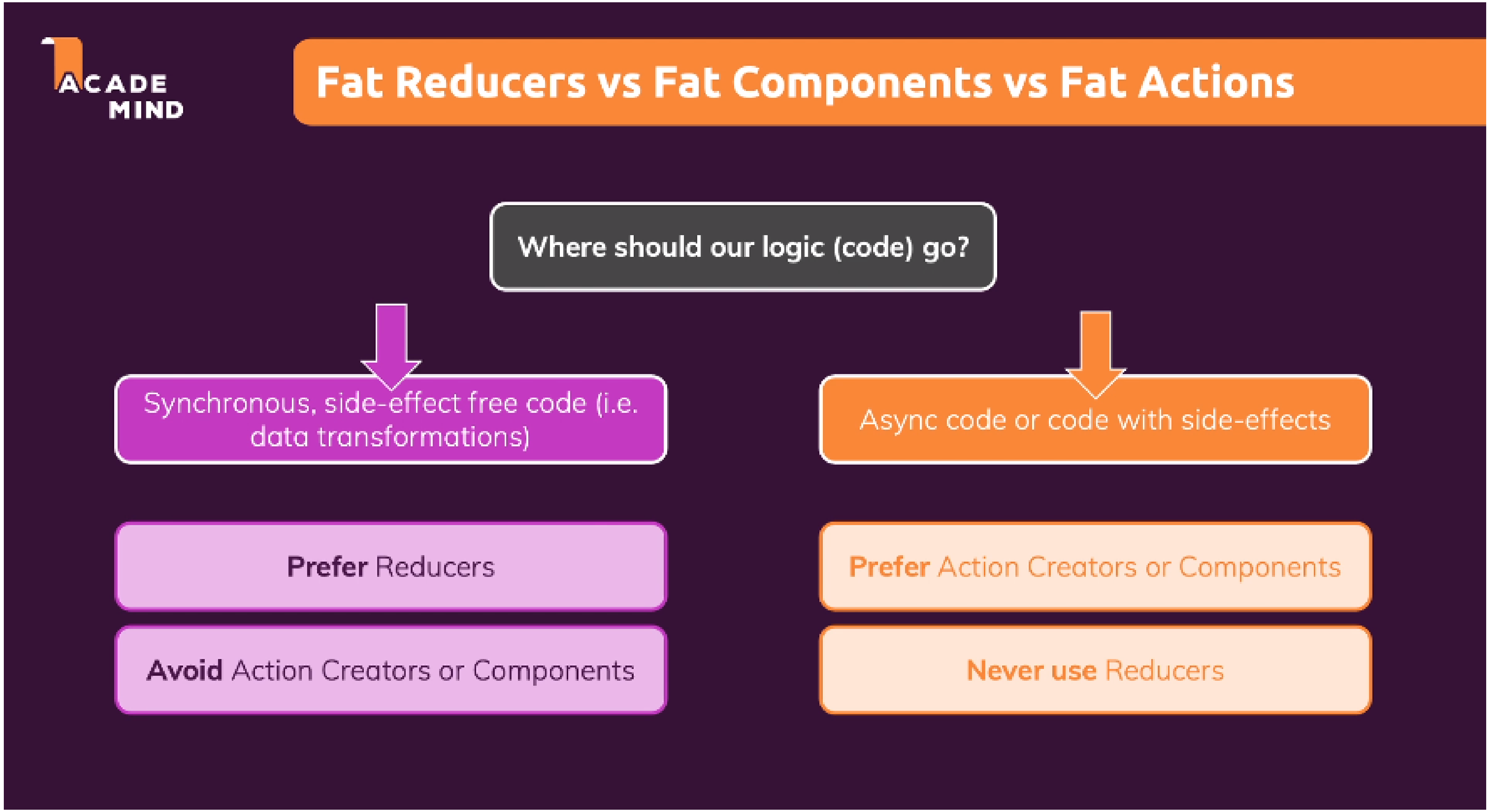
Does a lot of work
(i.e. transforms data + stores data)

Send data & receive + use response
(i.e. less code on the frontend, ahead of the reducer)

Does NOT a lot of work
(i.e. just stores incoming data)

Transform data & send data
(i.e. more code on the frontend, ahead of the reducer)

Frontend React App



Fat Reducers vs Fat Components vs Fat Actions

Where should our logic (code) go?

Synchronous, side-effect free code (i.e. data transformations)

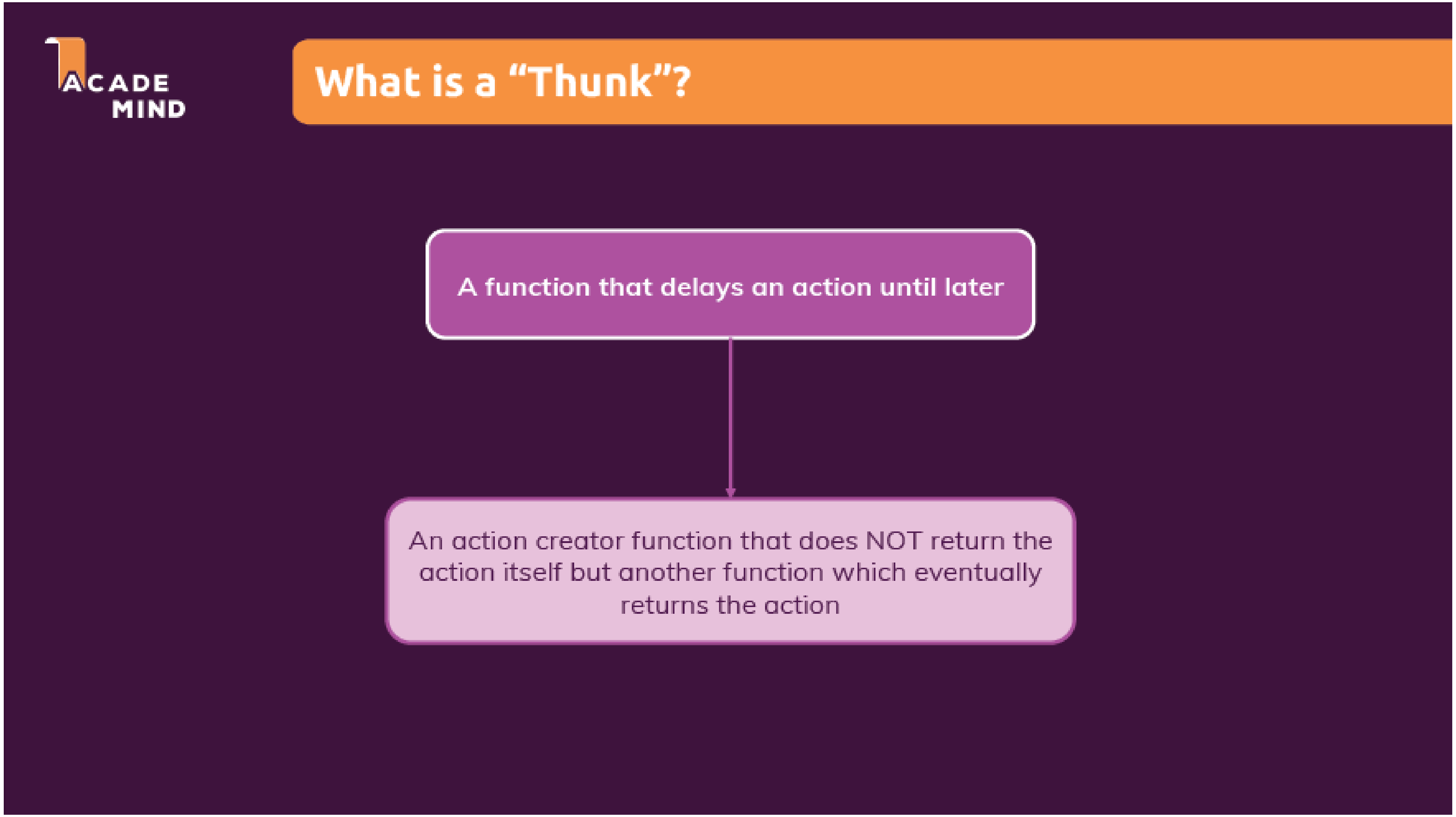
Prefer Reducers

Avoid Action Creators or Components

Async code or code with side-effects

Prefer Action Creators or Components

Never use Reducers



What is a “Thunk”?

A function that delays an action until later

An action creator function that does NOT return the action itself but another function which eventually returns the action