

Curso de **Vue.js** - Clase 6

Vuenos días! 🖐️



FICTIZIA



Vuex

State Management



MVC

Modelo – Vista – Controlador



JS & JQuery Style

Event Handler

Event Handler

Event Handler

Event Handler

Event Handler



DOM

DOM

DOM

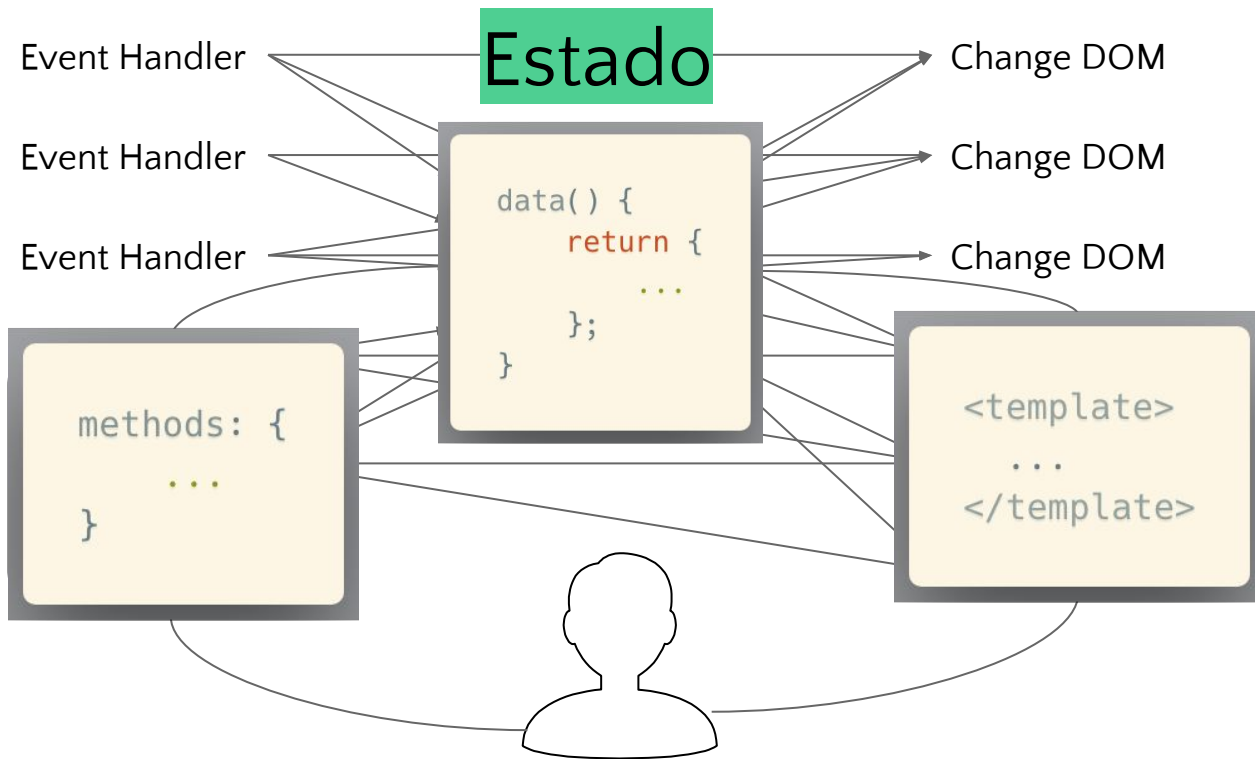
DOM

DOM

→ Change DOM

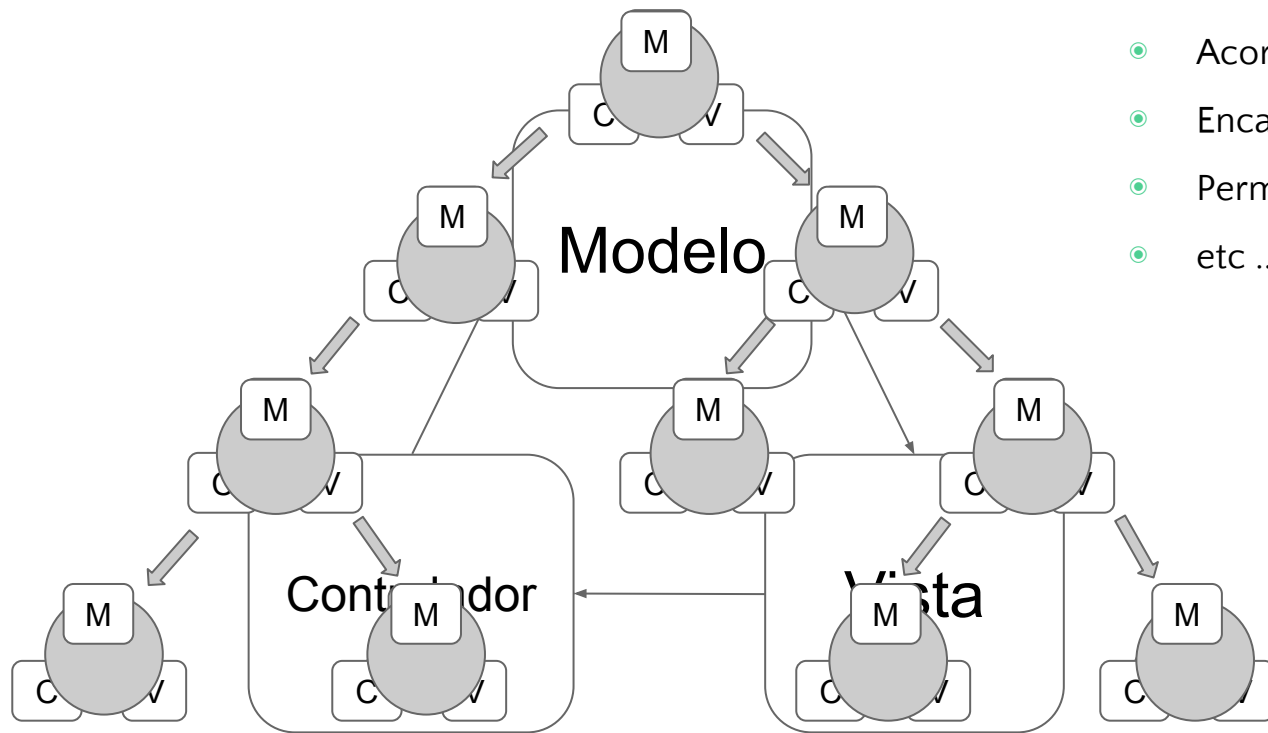


MVC Style

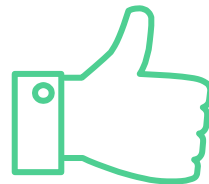




Components Style

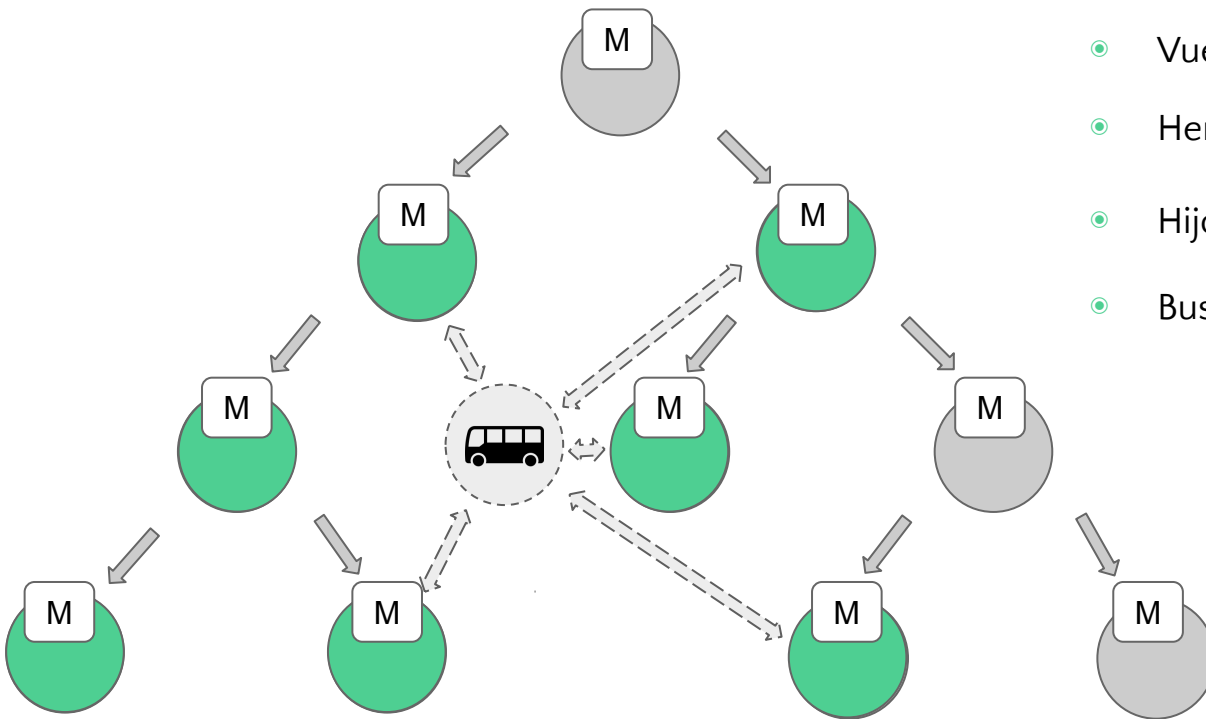


- Acorde con el DOM (HTML).
- Encapsula responsabilidad.
- Permite reusabilidad.
- etc ...





Comunicación



- Vue events & properties.
- Hermano-padre-hermano.
- Hijo-padre-abuelo.
- Bus de eventos.

Events party!



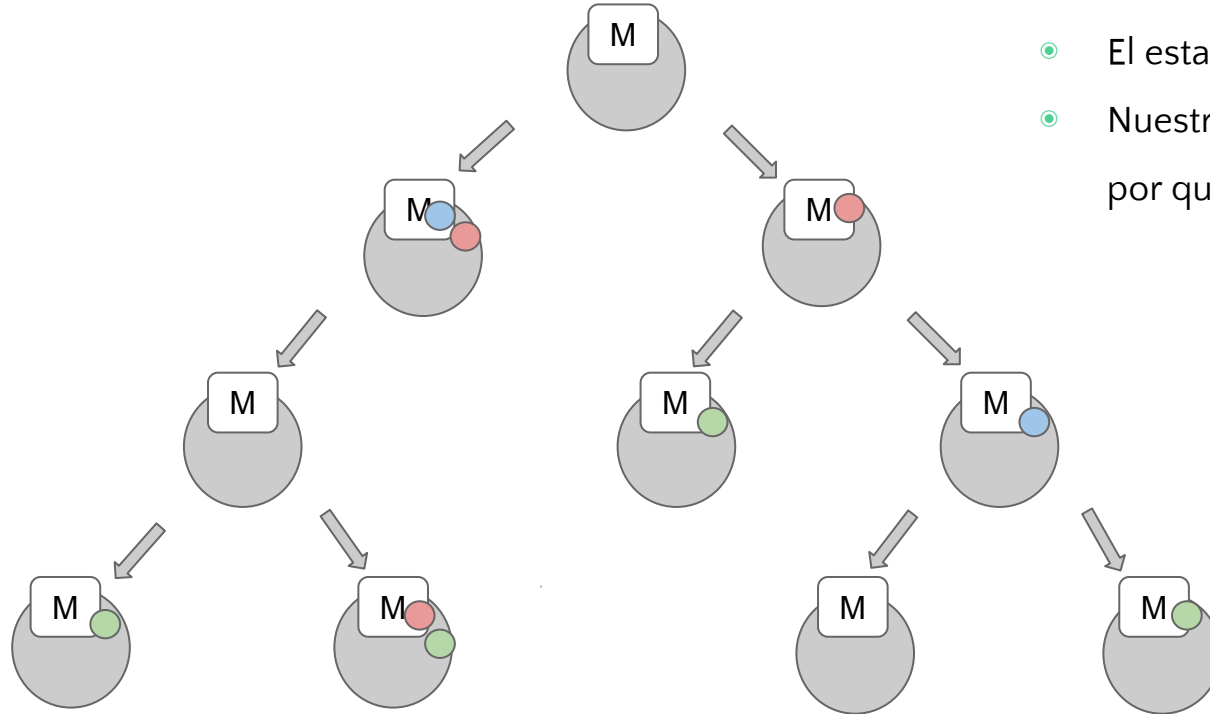


State management

Stat... ¿qué?



Gestión del estado



- El estado se encuentra distribuido.
- Nuestro árbol de estados no tiene por qué ser igual al DOM.



¿Estado?

- Suena complicado, es algo abstracto.
- Se manipula de forma intuitiva programando.
- Es fácil despreocuparse por él.
- Los problemas suelen surgir cuando ya es “demasiado” tarde.





Vuex

¡Al rescate!

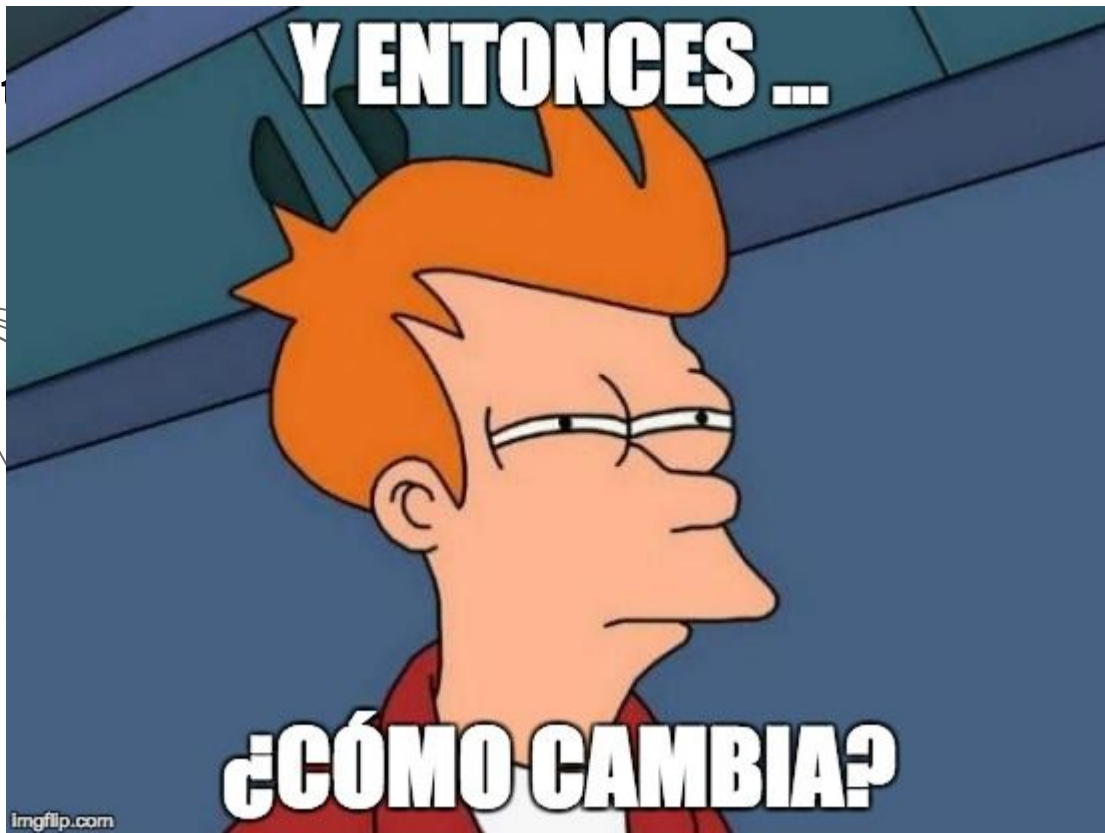


Sta

¿Variable global?



M



estado se encuentra
distribuido.

en único estado global.

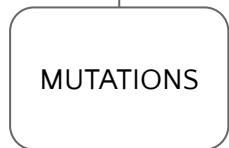
estado es de solo
estructura.



Mu



mutate



tado global.

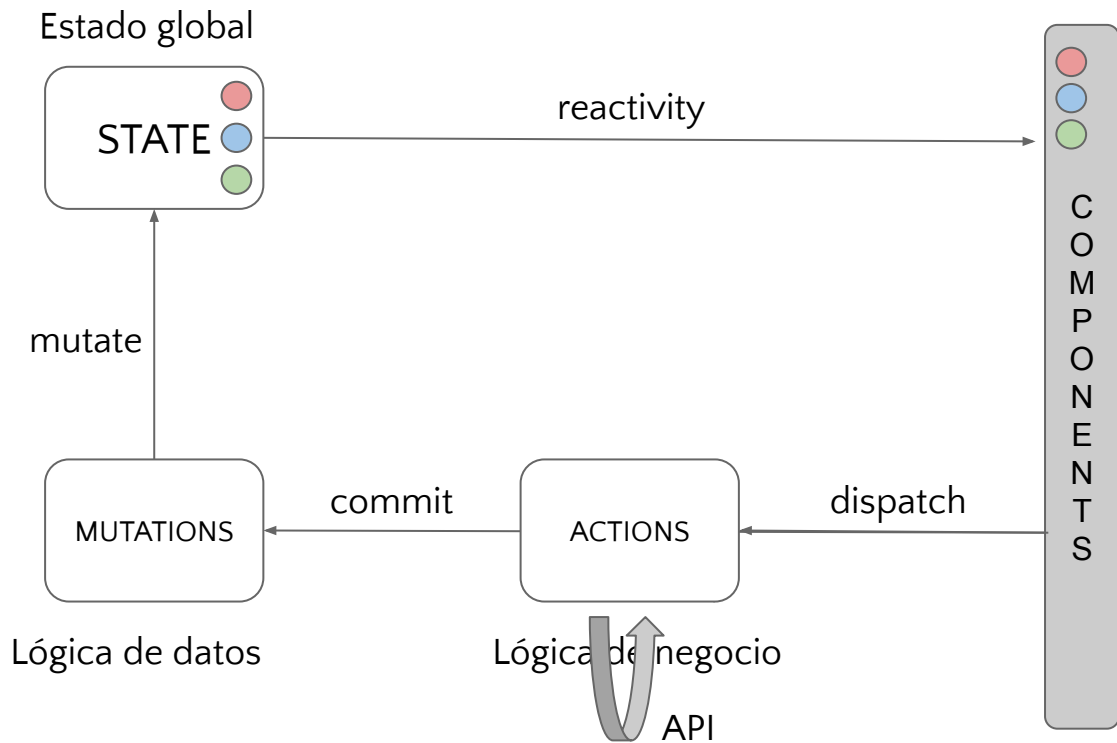
s de solo lectura.

lo cambia por las

ones son síncronas.



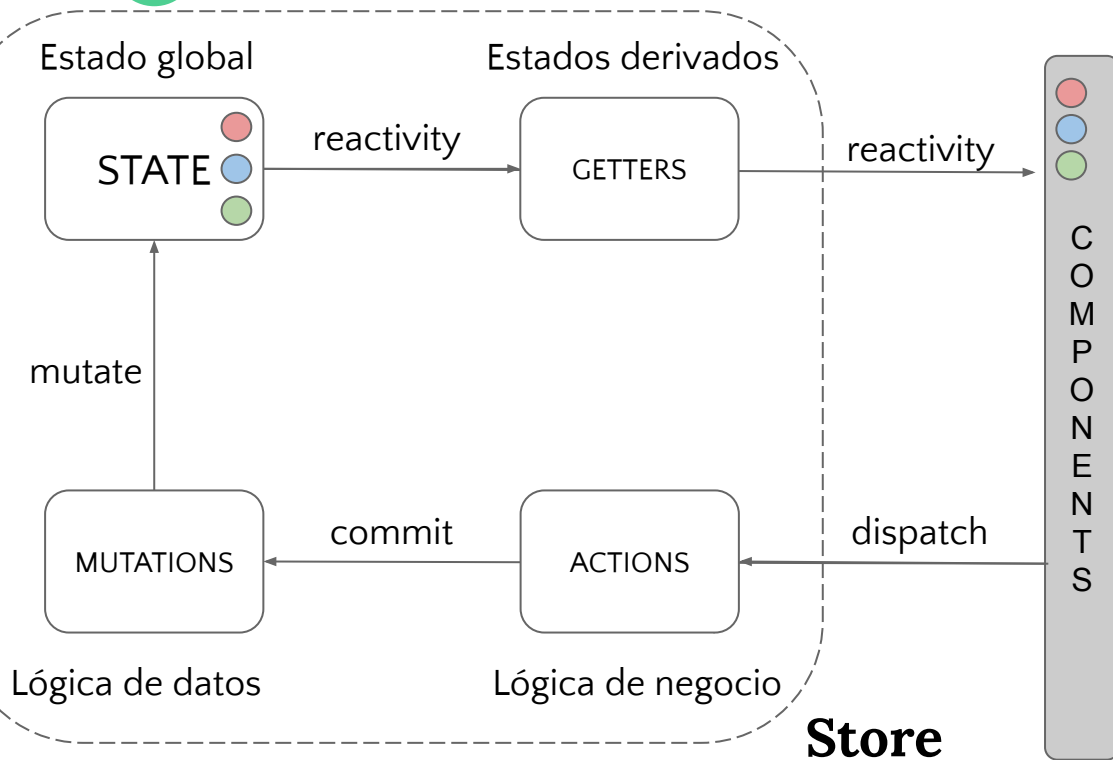
Actions



- Un único estado global.
- El estado es de solo lectura.
- El estado solo cambia por las mutaciones.
- Las mutaciones son síncronas.
- Las acciones pueden ser asíncronas.
- Las acciones se pueden componer.



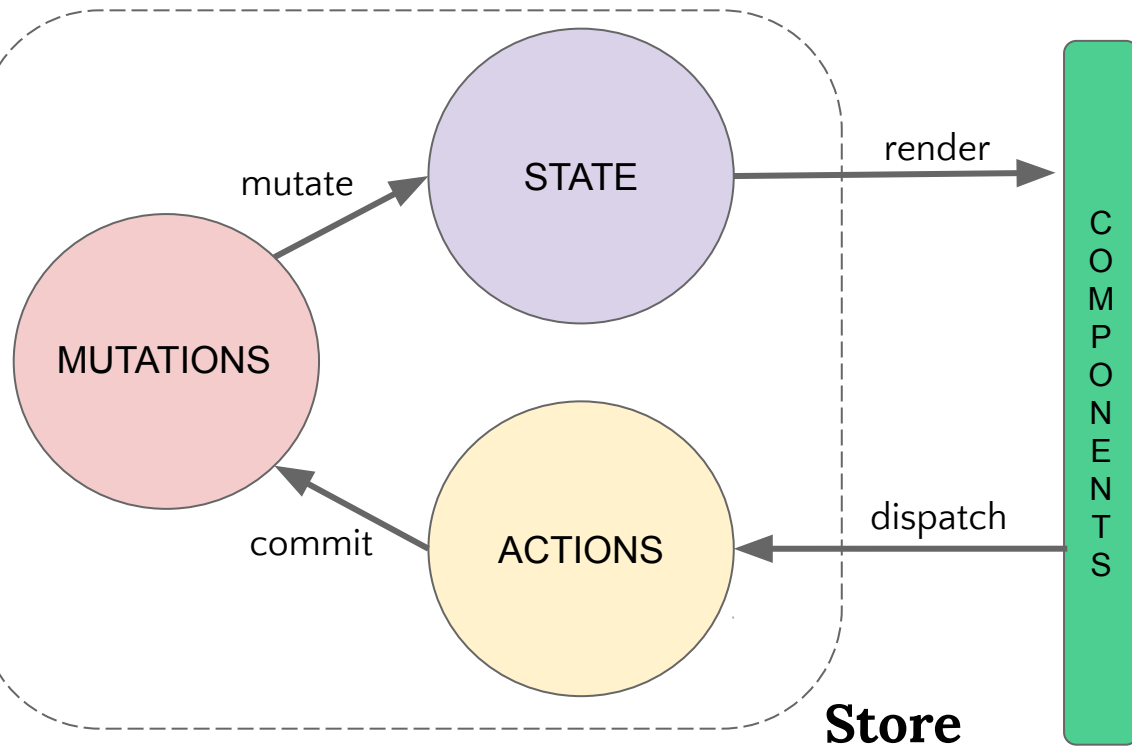
Vuex - Getters



- Un único estado global.
- El estado es de solo lectura.
- El estado solo cambia por las mutaciones.
- Las mutaciones son síncronas.
- Las acciones pueden ser asíncronas.
- Las acciones se pueden componer.



Vuex - Store



- Un único estado global de sólo lectura.
- El estado solo cambia por mutaciones síncronas.
- Las acciones generan mutaciones y pueden ser asíncronas.



Ventajas extras

- Te hace pensar sobre el estado.
- Te añade 2 capas de abstracción (negocio y datos).
- Evita “hacer de puente” entre dos componentes.
- De repente tienes una “caché” :O
- Las stores son modulables de forma fractal.
- Permite HMR con webpack.
- Permite Time-traveling con las [Vue devtools](#).



Vuenas prácticas

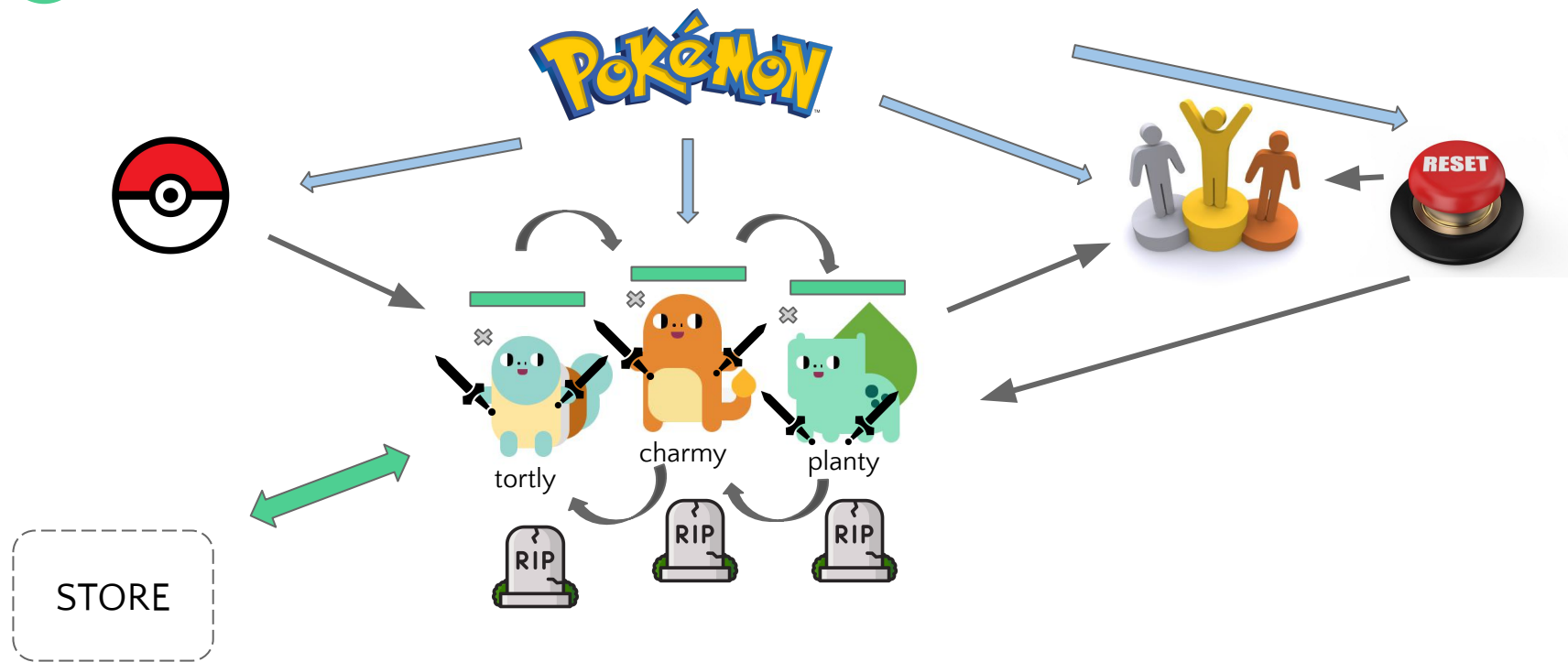
- El estado debe modelarse pensando sólo en los datos y evitando la redundancia, los getters pueden servirlo de forma más cómoda.
- Puede seguir existiendo estado local y comunicación padre-hijo, cosas puramente UI suelen ser locales.
- Los componentes solo deberían “dispatchear” acciones, no “commitear” mutaciones.
- Las mutaciones deben centrarse sólo en manipular los datos, las acciones pueden llevar la lógica de negocio.
- Leerse los [change detection caveats](#) de Vue ya que también se aplican a Vuex.

**Talk is cheap.
Show me the code.**

Linus Torvalds



Un ejemplo



const
at

const

A

};

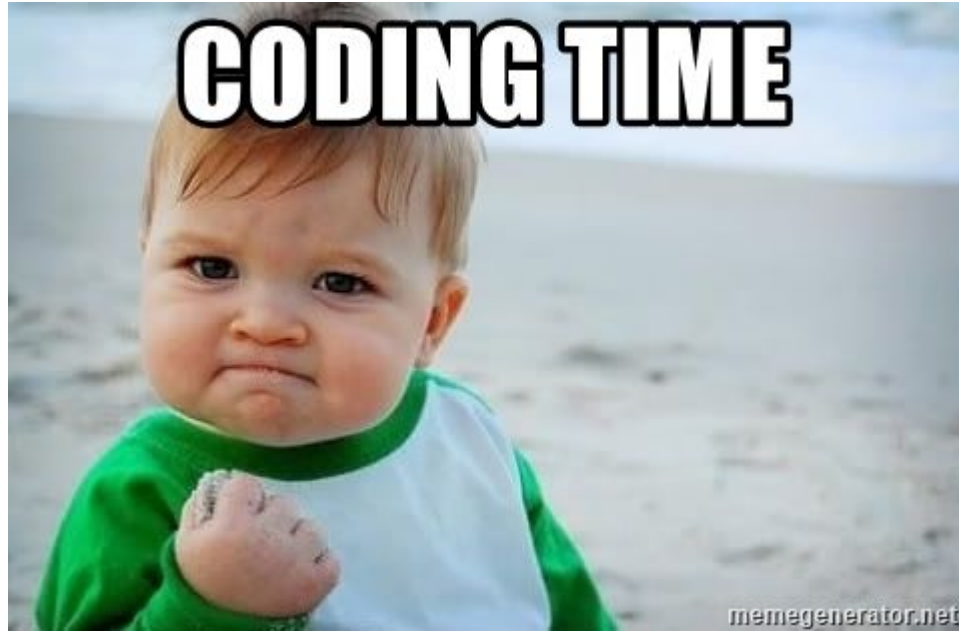
},
};



```
const state = {  
  pokemons: {  
    charmy: {  
      type: 'FIRE',  
      name: 'charmy',  
      life: 100,  
      attack: 10,  
      dying: false,  
    },  
    planty: {  
      type: 'PLANT',  
      name: 'planty',  
      life: 130, // 150  
      attack: 8,  
      dying: false,  
    },  
  },  
};
```

```
ANT) ||  
ATER) ||  
IRE)
```

oner.



<https://github.com/rubnvp/vuex-example/tree/initial>

Vuex: G
¿para qué



Nos **Vu**emos! 🖐️



FICTIZIA

Curso de Vue.js, clase 6, 31 de octubre de 2019