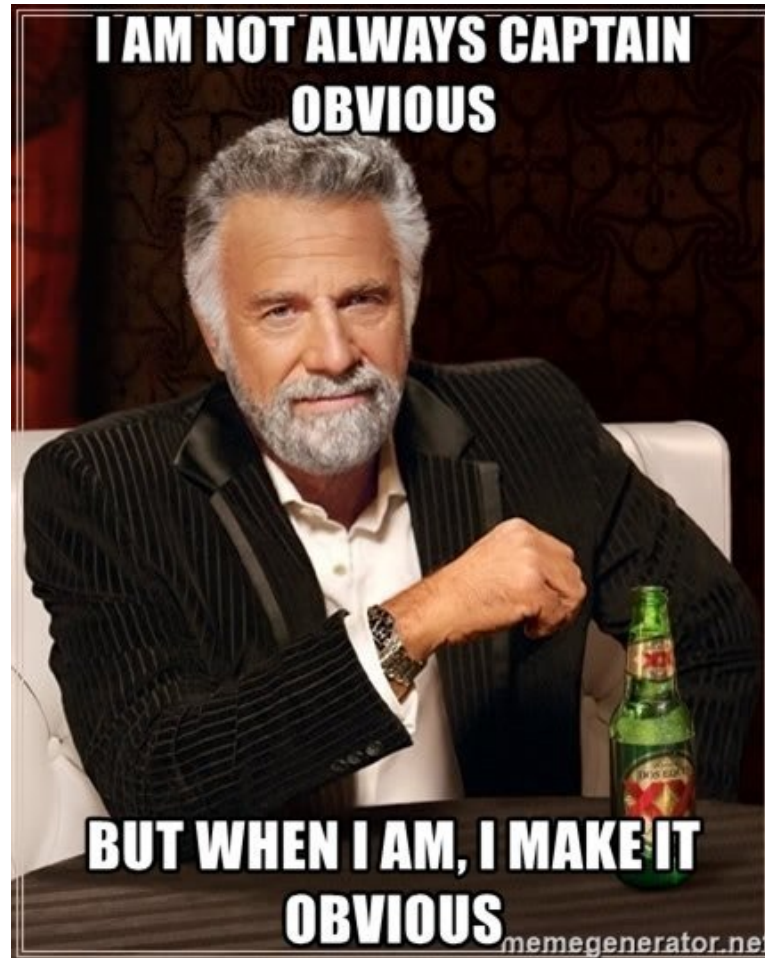


Programmation web - Client riche

Les événements

Un événement, c'est quand il se passe quelque chose...



... sur un élément

- `click`
- `mouseover` / `mouseout`
- `mousedown` / `mouseup`
- `mousemove`
- `focus` / `blur`
- `submit`
- `keydown` / `keyup`

... sur le document

- `DOMContentLoaded`

... sur le style d'un élément

- `transitionend`

... sur d'autres choses

```
function handler() {  
    console.log("You clicked!")  
}
```

```
element.addEventListener(  
    "click", // Type de l'événement  
    handler // Fonction à exécuter  
)
```



[https://codesandbox.io/embed/r0565p61op?
autoresize=1&hidenavigation=1](https://codesandbox.io/embed/r0565p61op?autoresize=1&hidenavigation=1)



[https://codesandbox.io/embed/o89q2o3lq?
autoresize=1&hidenavigation=1](https://codesandbox.io/embed/o89q2o3lq?autoresize=1&hidenavigation=1)



[https://codesandbox.io/embed/vn4rnjp140?
autoresize=1&hidenavigation=1](https://codesandbox.io/embed/vn4rnjp140?autoresize=1&hidenavigation=1)

L'objet Event


L'objet Event

Lorsqu'un événement se produit, le navigateur crée un objet Event dans lequel il place des détails sur celui-ci :

- Type de l'événement
- Élément sur lequel il a été déclenché
- Coordonnées de la souris au moment du click
- Touche appuyée ou relâchée
- ...



[https://codesandbox.io/embed/k900m1l19r?
autoresize=1&hidenavigation=1](https://codesandbox.io/embed/k900m1l19r?autoresize=1&hidenavigation=1)



[https://codesandbox.io/embed/n9x6z74mvp?
autoresize=1&hidennavigation=1](https://codesandbox.io/embed/n9x6z74mvp?autoresize=1&hidennavigation=1)

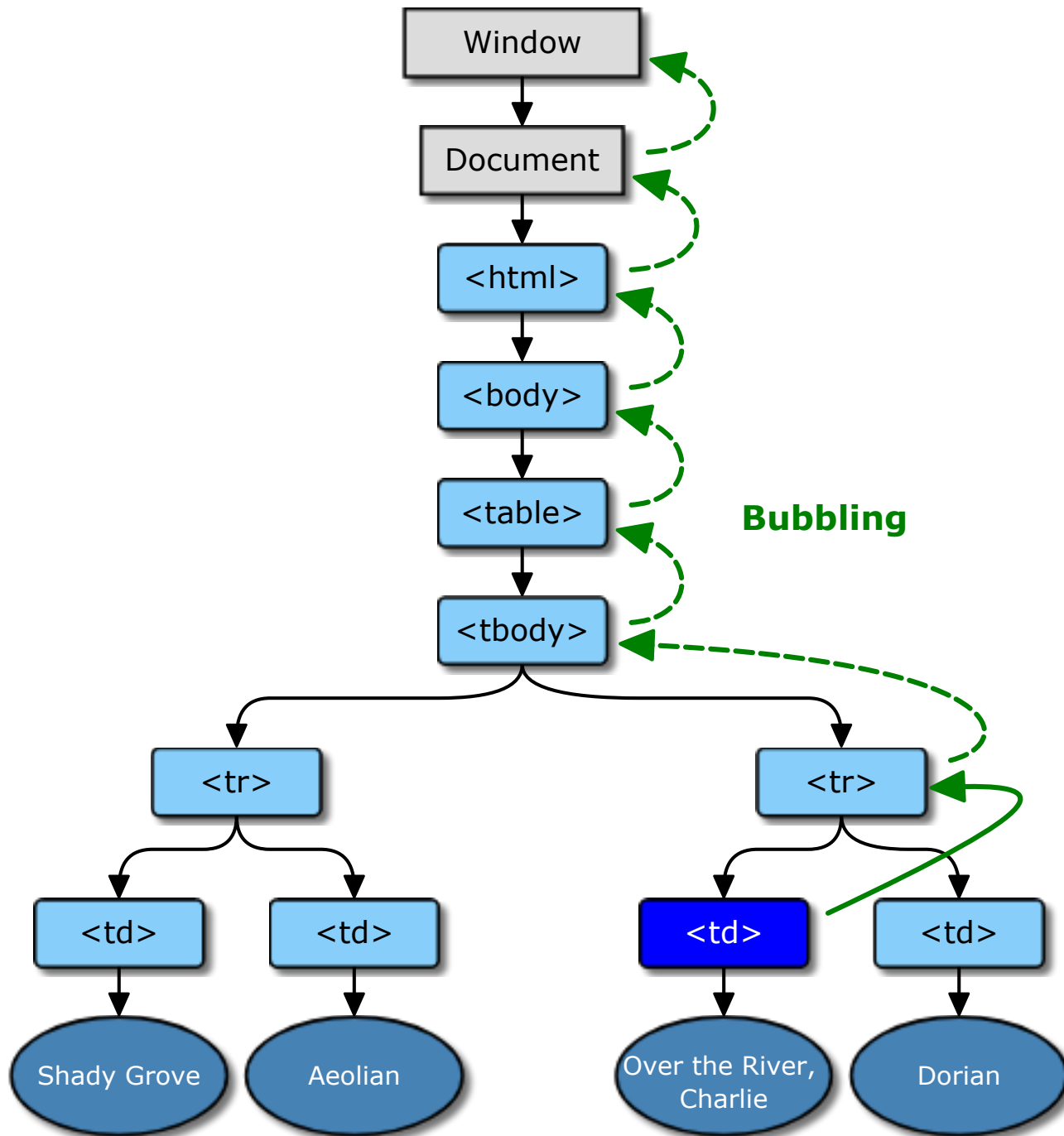


[https://codesandbox.io/embed/jpxylj0jj3?
autoresize=1&hidenavigation=1](https://codesandbox.io/embed/jpxylj0jj3?autoresize=1&hidenavigation=1)

Les phases d'un événement

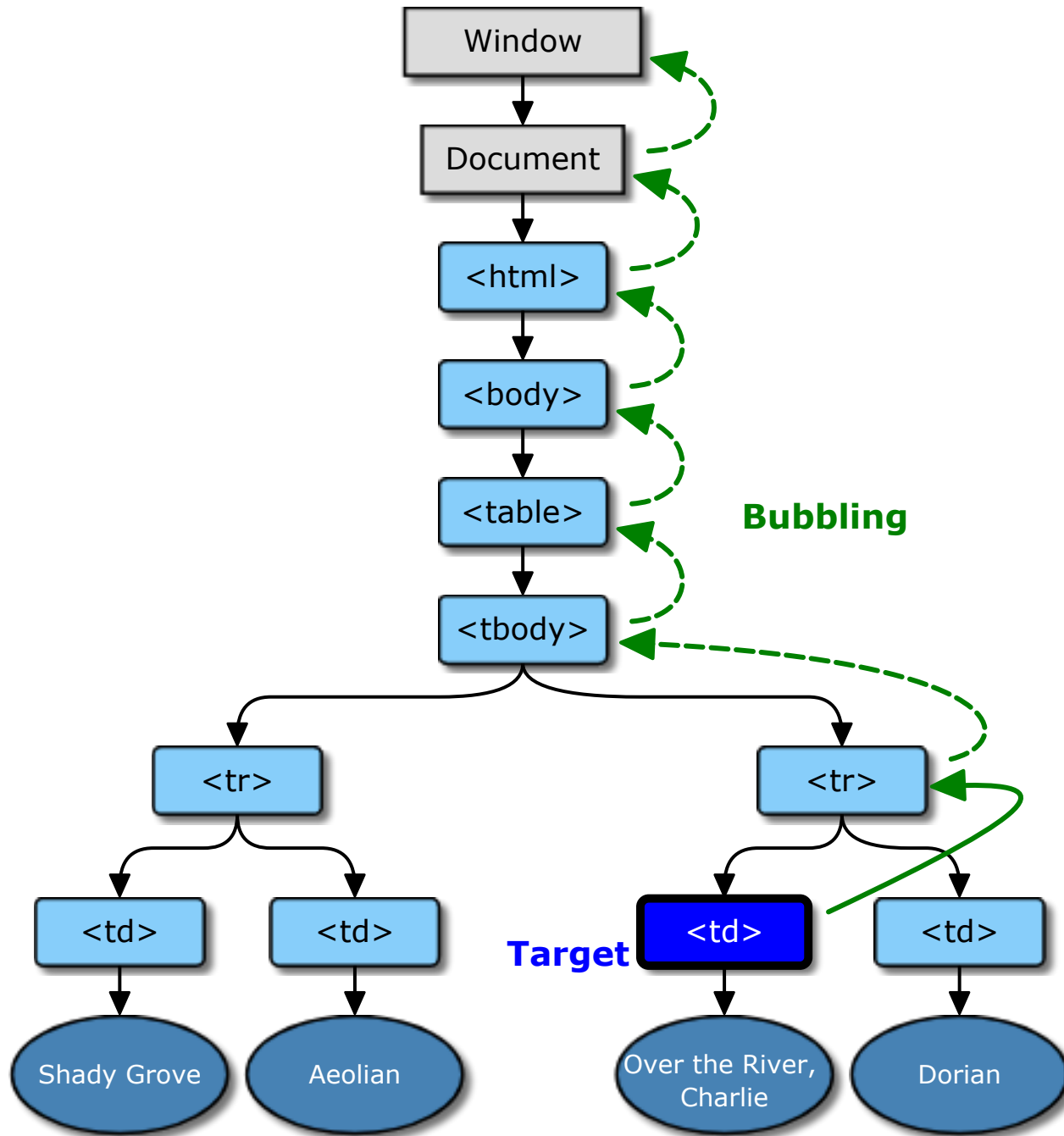


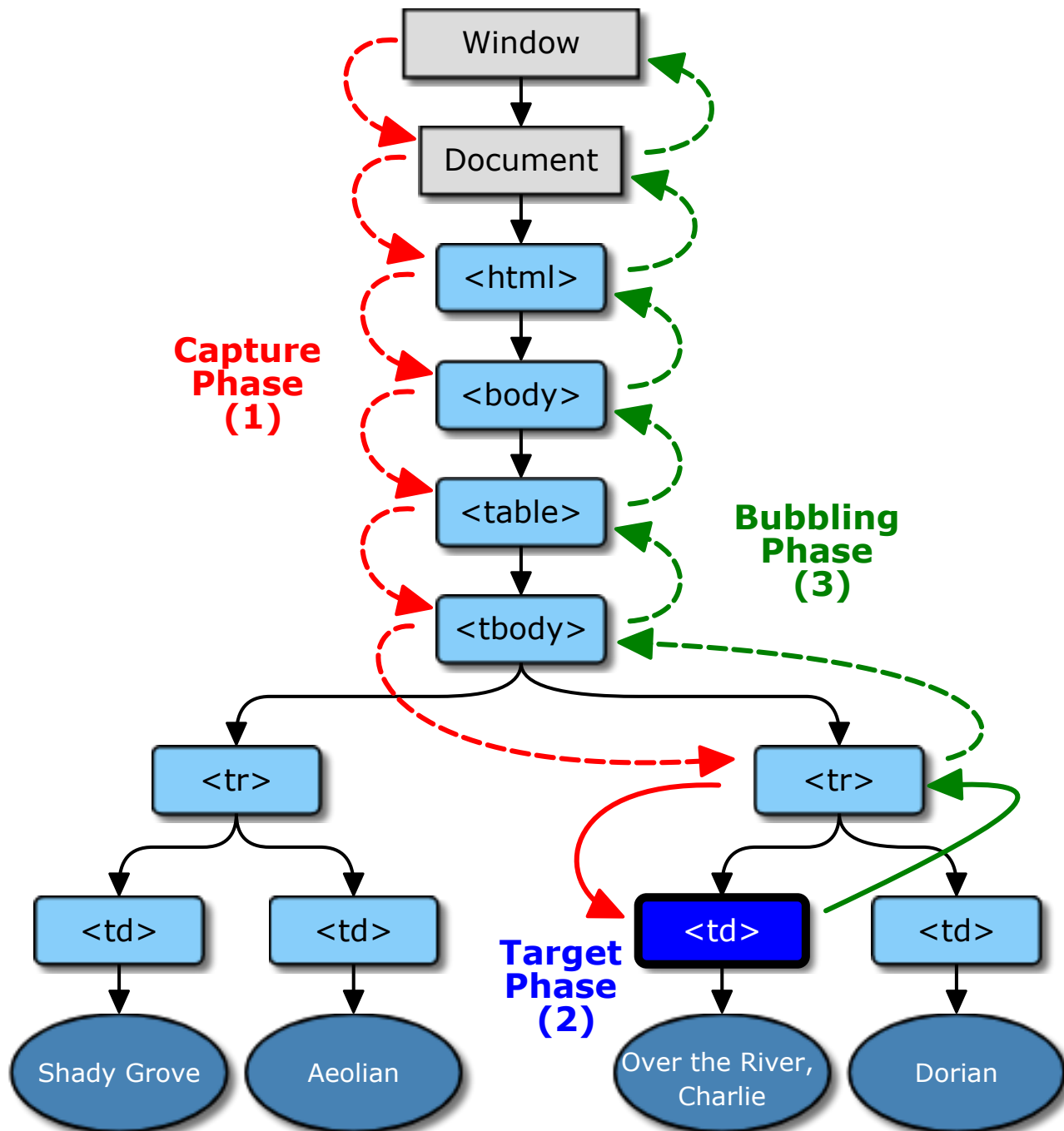
[https://codesandbox.io/embed/2vm9jjovoj?
autoresize=1&hidenavigation=1](https://codesandbox.io/embed/2vm9jjovoj?autoresize=1&hidenavigation=1)





[https://codesandbox.io/embed/9ko58ol0y?
autoresize=1&hidenavigation=1](https://codesandbox.io/embed/9ko58ol0y?autoresize=1&hidenavigation=1)







[https://codesandbox.io/embed/o9272wx55z?
autoresize=1&hidenavigation=1](https://codesandbox.io/embed/o9272wx55z?autoresize=1&hidenavigation=1)

La délégation d'événements



[https://codesandbox.io/embed/2onw79r6zr?
autoresize=1&hidenavigation=1](https://codesandbox.io/embed/2onw79r6zr?autoresize=1&hidenavigation=1)



[https://codesandbox.io/embed/jzwmj5w9zyv?
autoresize=1&hidenavigation=1](https://codesandbox.io/embed/jzwmj5w9zyv?autoresize=1&hidenavigation=1)



[https://codesandbox.io/embed/k55m6603x3?
autoresize=1&hidenavigation=1](https://codesandbox.io/embed/k55m6603x3?autoresize=1&hidenavigation=1)



La délégation d'événement est un pattern parfait lorsque :

- *des éléments dynamiques (e.g. qui apparaissent, disparaissent, changent de position...) sont tous contenus dans un même élément*
- *l'élément qui contient les autres n'est pas lui-même dynamique*



*Implémenter la délégation d'événement
proprement en gérant tous les cas à la marge
peut être fastidieux. [ftdomdelegate](#) gère tout ça
très bien*

Des questions ?



TD