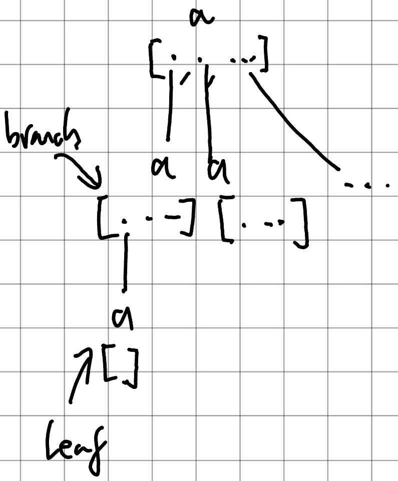
Dynamic Signal graph

widget < container / component ~ | Branch / Leaf

usual formulation of DOM Tree : Annotation Tree (CoFree)

```
data CoFree f a = Tuple a f (CoFree f a)

type DOM a = CoFree List a
```



"every node may/not contain a list of nodes"

Data Layer

- structured as a graph of entities
- may contain ontology of entities (through secondary indices)

how to access focused pieces of data?

$\boxed{\text{query}}$

View Layer

organized around the partition of event space (edge graph)

view ought to have as less knowledge about how data are organized in (partial
Data Layer order
as possible

⇒ resistent to reorganization of data

Angular, Vue

- dual-way data binding

$\xrightarrow{\text{jump}}$

Elm

- directional data binding

fragmented designs,

independent, local

- local state management
- local/store-driven communication/coordination

- global state management (single state store)

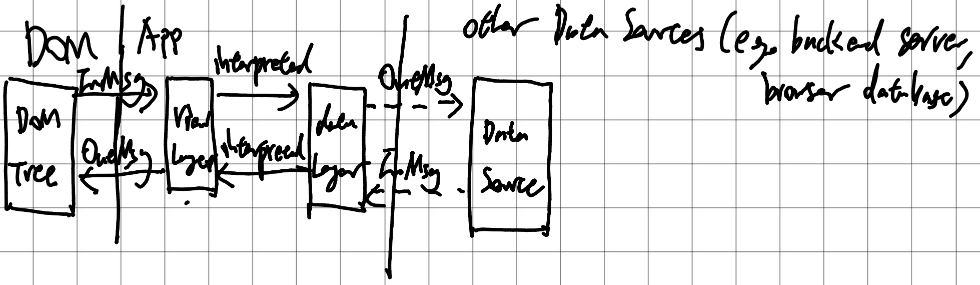
---

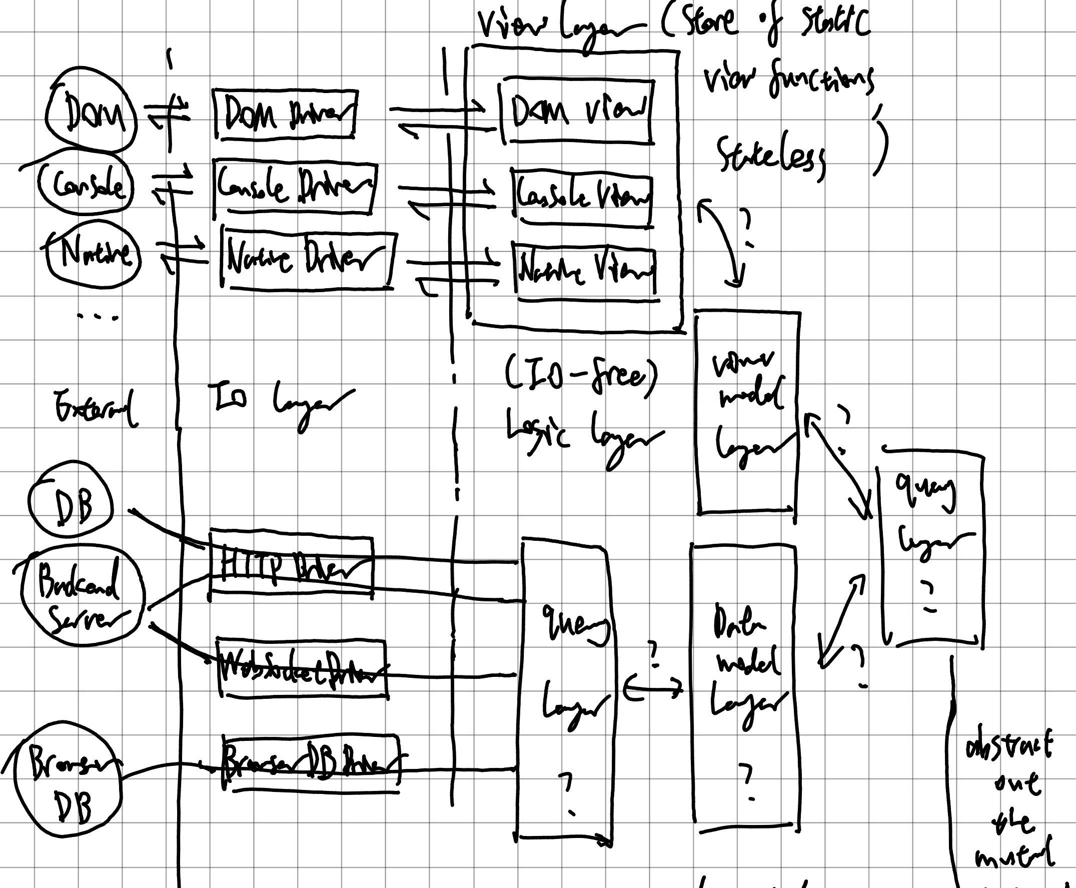
primitive events
- state update of stateful (~~view~~ input) component in DOM
- triggering of stateless (input) component in DOM.

dual-way allow direct access to external state in DOM.

directional external state update as event propagates the signal graph.

binding source forces updates of sinks.





never decoupled,
loosely coupled through a mediator