Tape Framework

Less random stuff in random places

clyfe

December 11, 2020

Modules

- (Ported from Duct.)
- The configuration is initiated twice.
- First into an intermediate configuration,
- which in turn is initiated into the system.

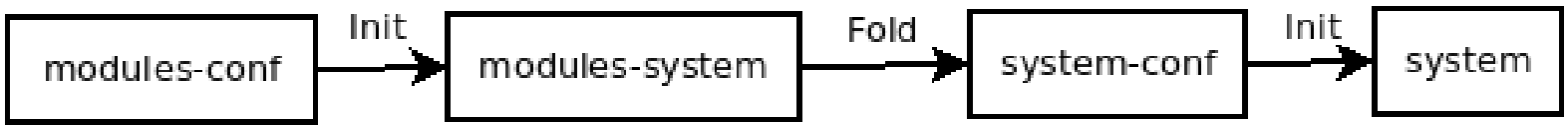

Figure: Modules

Components

- Note: two types of components:
  - Module components: fns that add to the config map.
  - System components: whatever.

```
(defmethod ig/init-key :tape/const [_ v] v)
```

Figure: Need later

Re-Frame Globals

- Re-Frame chose globals for a simpler API.

```
(ns my.app.p.controller)
(rf/reg-event-db ::index
  (fn [db _]
    {:people [...]}))
```

Figure: Re-Frame plain

```
(ns my.app.people.view)
(defn index []
  (for [person people]
    [:p person]))
```

Figure: Reagent plain

Use Integrant

- Begone globals.

```
(ns my.app.p.controller)
(defmethod ig/init-key
  ::index [_ _]
  (fn [db _]
    {:people [...]}))
```

Figure: Re-Frame Integrant

```
(ns my.app.people.view)
(defmethod ig/init-key
  ::index [_ _]
  (fn []
    (for [p peps] [:p p])))
```

Figure: Reagent Integrant

Automate use Integrant

- Leverage Integrant keys inheritance.

```
(ns my.app.p.controller)
(defn index [db _]
  {:people [...]})
(derive ::index
        :tape/const)
```

Figure: Re-Frame Integrant

```
(ns my.app.people.view)
(defn index []
  (for [p peeps] [:p p]))
(derive ::index
        :tape/const)
```

Figure: Reagent plain

Use Modules

- Modules are buckets of handlers, subscritptions, view fns.

```
;; cont'd from above
(defmethod ig/init-key
  ::module [_ _]
  (fn [config]
    (merge config
           {::index deps})))
```

Figure: Controller module

```
;; cont'd from above
(defmethod ig/init-key
  ::module [_ _]
  (fn [config]
    (merge config
           {::index deps}
```

Figure: View module

Automate use Modules

- Have to annotate fns.

```
(c/defmodule)

;; inspects ns
;; expands into
;; module above
```

Figure: Controller module

```
(v/defmodule
 my.app.p.controller)
;; inspects ns
;; expands into
;; module above
```

Figure: View module

Leverage Intuitions

- MVC is well understood, let's use it in naming.
- Controller: Re-Frame stuff.
- View: Reagent stuff.
- Model: whatever.

Consistent Naming

- Force: event names to be namespaced.
- Force: handlers names to match event names.
- If a view is rendered as a result of an event handler executing, their names should match.

```
:my.app.people.controller/index  ;;  event
my.app.people.controller/index   ;;  handler
my.app.people.view/index         ;;  view
```

Leverage Metadata

- Annotate fns.
- Automate module definition via defmodule.

```
(ns my.app.p.controller)
(defn ::c/event-db
  index [db _]
  {:people [...]})

(c/defmodule)
```

Figure: Controller module

```
(ns my.app.people.view)
(defn ::v/view
  index []
  (for [p peps] [:p p]))
(v/defmodule
 my.app.p.controller)
```

Figure: View module

Links

https://github.com/tape-framework