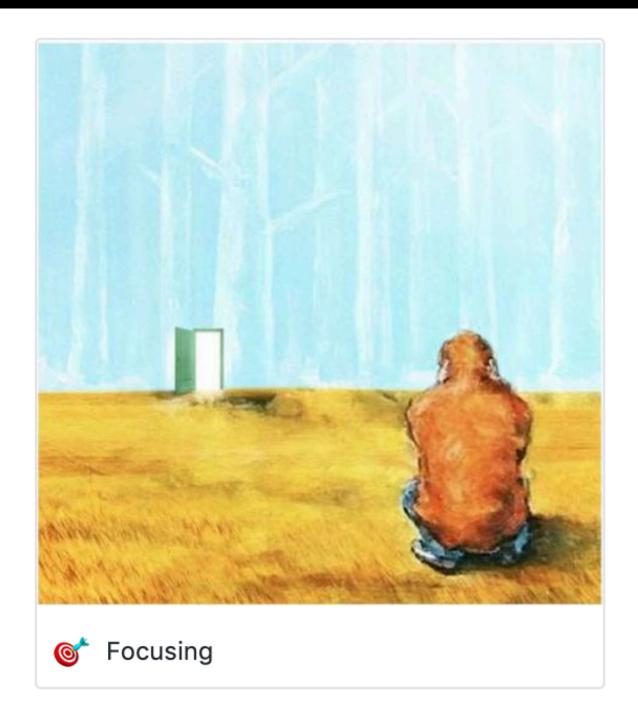
LiveView香不香?

你真的不用写JS了吗

关于我

• Name: 柴千祥

• Github: https://github.com/caicaishmily



chaichai

caicaishmily

Edit profile

认真,是我们改变世界的唯一方式

2 @letotecn

- shenzhen
- □ davecqx@gmail.com
- https://caicaishmily.github.io/

刊售提要

- WebSocket
- LiveView

WebSocket

WebSocket 是 HTML5 开始提供的一种在单个 TCP 连接上进行全双工通讯的协议。

在 WebSocket API 中,浏览器和服务器只需要完成一次握手,两者 之间就直接可以创建持久性的连接,并进行双向数据传输。

能更好的节省服务器资源和带宽,并且能够更实时地进行通讯

LiveView

Phoenix LiveView is an exciting new library which enables rich, real-time user experiences with server-rendered HTML. LiveView powered applications are stateful on the server with bidrectional communication via WebSockets, offering a vastly simplified programming model compared to JavaScript alternatives.

By: Chris McCord • December 12, 2018



JS Demo

```
<template>
 <div id="app">
   <h3 v-if="loading">Loading...</h3>
   v-for="item in girls" :key="item.avatar">
       <img style="width: 100%;" :src="item.avatar" />
       <div class="list-footer">
         <span>{{ item.name }}</span>
         <button @click="like(item.id)" :id="item.id">Like
           <span style="margin-left: 5px;">{{item.like}}</span>
         </button>
       </div>
     </div>
</template>
```

```
export default {
 name: 'app',
 data() {
   return {
     loading: true,
     test: "test",
     girls: []
 methods: {
   async like(id) {
     let result = await this.$apollo.mutate({
       mutation: gql`mutation ($id: Float!) {
          like(id: $id) {
           id
           age
           like
           name
           avatar
           dislike
       }`,
        variables: {
         id: Number.parseInt(id),
       },
     this.girls = this.girls.map(item => item.id == id ? result.data.like : item).sort(item => item.id)
 async mounted() {
   let result = await this.$apollo.query({
     query: gql`query {
       girls{
         age
         id
         name
         avatar
         like
         dislike
    })
   this.girls = result.data.girls.sort(item => item.id)
   this.loading = false
```

LiveView Code Live

```
<section>
 <%= for item <- @image_list do %>
     style="padding: 10px 10px 0; background: #eee; border-radius: 5px;">
       <img src="<%= item.avatar %>" alt="piaoxinhui"/>
       <div style="display: flex; justify-content: space-between; align-items: center;">
        <span><%= item.name %></span>
         <button phx-click="like" phx-value-id="<%= item.id %>">Like <span><%= item.like %></span></button>
       </div>
     <% end %>
 </section>
```

```
def mount(_session, socket) do
  LivePhoenixExampleWeb.Endpoint.subscribe(@topic)
  {:ok, assign(socket, image_list: get_image_list())}
end
@spec handle_event(<<_::32>>, map, Phoenix.LiveView.Socket.t()) :: {:noreply, any}
def handle_event("like", %{"id" => id}, socket) do
  with {:ok, _girl} <- update_like(id) do</pre>
    image_list = get_image_list()
    LivePhoenixExampleWeb.Endpoint.broadcast_from(self(), @topic, "like", %{image_list: image_list})
    {:noreply, assign(socket, image_list: image_list)}
  else
    {:error, error} ->
      {:noreply, assign(socket, %{error: error})}
  end
end
@spec handle_info(%{event: <<_::32>>, payload: maybe_improper_list | map}, Phoenix.LiveView.Socket.t()) :: {:noreply, any}
def handle_info(%{event: "like", payload: state}, socket) do
  {:noreply, assign(socket, state)}
end
```

更多关于LiveView

https://phoenixphrenzy.com/results

https://dockyard.com/blog/categories/elixir

https://github.com/phoenixframework/phoenix_live_view

https://elixirschool.com/blog/

https://flappy-phoenix.herokuapp.com/game