

```

1 defmodule ProjetinhoWeb.PageLive do
2   use ProjetinhoWeb, :live_view
3
4   @colors ["pink", "red", "purple", "green", "blue"]
5
6   def mount(_params, _session, socket) do
7     if connected?(socket) do
8       :timer.send_interval(1000, :tick)
9     end
10
11     {:ok, assign(socket, x: 0, y: 0, color: "purple")}
12   end
13
14   def render(assigns) do
15     ~L"""
16     <section phx-window-keydown="keydown" class="phx-hero">
17       <pre><%= inspect({@x, @y}) %></pre>
18       <svg width="700" height="350">
19         <rect x="<%= @x * 25 %>" y="<%= @y * 25 %>"
20           width="25" height="25" style="fill:<%= @color %>;" />
21       </svg>
22       <pre><%= inspect(@color) %></pre>
23     </section>
24     """
25   end
26
27   def handle_info(:tick, socket) do
28     {:noreply, assign(socket, color: Enum.random(@colors))}
29   end
30
31   def handle_event("keydown", %{"key" => "ArrowRight"}, socket) do
32     {:noreply, update(socket, :x, &(&1 + 1))}
33   end
34
35   def handle_event("keydown", %{"key" => "ArrowLeft"}, socket) do
36     {:noreply, update(socket, :x, &(&1 - 1))}
37   end
38
39   def handle_event("keydown", %{"key" => "ArrowUp"}, socket) do
40     {:noreply, update(socket, :y, &(&1 - 1))}
41   end
42
43   def handle_event("keydown", %{"key" => "ArrowDown"}, socket) do
44     {:noreply, update(socket, :y, &(&1 + 1))}
45   end
46
47   def handle_event("keydown", _key, socket) do
48     {:noreply, socket}
49   end
50 end

```

Mount

Render

Handle

# Mount

```
def mount(_params, _session, socket) do
  if connected?(socket) do
    :timer.send_interval(1000, :tick)
  end

  {:ok, assign(socket, x: 0, y: 0, color: "purple")}
end
```