```elixir
defmodule ProjetinhoWeb.PageLive do
  use ProjetinhoWeb, :live_view

  @colors ["pink", "red", "purple", "green", "blue"]

  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

  def render(assigns) do
    ~L"""
    <section phx-window-keydown="keydown" class="phx-hero">
      <pre><%= inspect({@x, @y}) %></pre>
      <svg width="700" height="350">
        <rect x="<%= @x * 25 %>" y="<%= @y * 25 %>"
        width="25" height="25" style="fill:<%= @color %>;" />
      </svg>
      <pre><%= inspect(@color) %></pre>
    </section>
    """
  end

  def handle_info(:tick, socket) do
    {:noreply, assign(socket, color: Enum.random(@colors))}
  end

  def handle_event("keydown", %{"key" => "ArrowRight"}, socket) do
    {:noreply, update(socket, :x, &(&1 + 1))}
  end

  def handle_event("keydown", %{"key" => "ArrowLeft"}, socket) do
    {:noreply, update(socket, :x, &(&1 - 1))}
  end

  def handle_event("keydown", %{"key" => "ArrowUp"}, socket) do
    {:noreply, update(socket, :y, &(&1 - 1))}
  end

  def handle_event("keydown", %{"key" => "ArrowDown"}, socket) do
    {:noreply, update(socket, :y, &(&1 + 1))}
  end

  def handle_event("keydown", _key, socket) do
    {:noreply, socket}
  end
end
```

Mount

Render

Handle

Mount

```elixir
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
```