```elixir
defmodule NameServer1 do
  @moduledoc """
  Very simple name server supporting two methods:
  add: Add a name and a place.
  find: Given a name, return the place or nil
  """

  def start(name) do
    Process.register(spawn(NameServer1, :loop, [name, %{}]), name)
    :ok
  end

  def rpc(name, request) do
    send(name, {self(), request})

    receive do
      {_name, response} -> response
    end
  end
```

The client and server parts.

Interface

Functions called by the process using the server.

```elixir
  def loop(name, state) do
    receive do
      {from, request} ->
        {response, newState} = handleRequest(request, state)
        send(from, {name, response})
        loop(name, newState)
    end
  end
```

Server

The server.

```elixir
  defp handleRequest({:add, name, place}, state) do
    newState = Map.put(state, name, place)
    {:ok, newState}
  end

  defp handleRequest({:find, name}, state) do
    {state[name], state}
  end
end
```

Implementation

Functions called by the server to implement the server logic.

What happens if a method call crashes?

- The whole server will crash! Not good

Lister: “I’m going to use my brains for the first time in my life.”
Kryten: “Considering the circumstances, sir, do you really believe that’s wise?”