```gleam
/// After the game code has been inputted on the join page
///
/// Sends the player to the `set_name_page`
///
pub fn on_join_game(
  message: String,
  player: PlayerSocket,
) -> WebsocketActorState {
  let assert Ok(juno.Object(message_dict)) = juno.decode(message, [])
  let assert Ok(juno.String(game_code)) = message_dict |> dict.get("gameCode")
  case int.parse(game_code) {
    Ok(code) -> {
      let assert Ok(waiting_games) = table.ref("waiting_games")
      case waiting_games |> table.lookup(code) {
        [] -> {
          let assert Ok(_) = mist.send_text_frame(player.socket, wrong_code())
          player.state
        }
        _ -> {
          waiting_games |> table.delete(code)
          process.send(
            player.state.director_subject,
            EnqueueParticipant(code, 0, player.state.ws_subject),
          )
          WebsocketActorState(..player.state, player: 0)
        }
      }
    }
    _ -> {
      let assert Ok(_) = mist.send_text_frame(player.socket, wrong_code())
      player.state
    }
  }
}
```