Demostración

∀ u :: Universo . ∀ o :: Objeto . elem o (objetos_en u) ⇒ elem (Right o) u

Hacemos inducción sobre el universo considerando
type Universo = [Either Personaje Objeto]

Caso base

u :: Universo . u = [] (conjunto vacío)

Qvq

∀ o :: Objeto . elem o (objetos_en []) ⇒ elem (Right o) [] ≡ (por definición de objetos_en)
∀ o :: Objeto . elem o [] ⇒ elem (Right o) [] ≡ (por definición de elem)
∀ o :: Objeto . False ⇒ False ≡ True ■

Caso inductivo

Hipótesis inductiva

∀ o :: Objeto . elem o (objetos_en xs) ⇒ elem (Right o) xs

Qvq

∀ x :: Either Personaje Objeto . ∀ o :: Objeto . elem o (objetos_en x:xs) ⇒ elem (Right o) x:xs

Por extensionalidad para Either basta con ver los casos (x = Left a) y (x = Right b)

Subcaso Left

∀ a :: Personaje . ∀ o :: Objeto . elem o (objetos_en (Left a):xs) ⇒ elem (Right o) (Left a):xs ≡ (por definición de objetos_en)

∀ a :: Personaje . ∀ o :: Objeto . elem o (map (\(Right x)-> x) (filter es_un_objeto (Left a):xs)) ⇒ elem (Right o) (Left a):xs ≡ (por definición de filter)

∀ a :: Personaje . ∀ o :: Objeto . elem o (map (\(Right x)-> x) (if es_un_objeto (Left a) then (Left a):(filter es_un_objeto xs) else (filter es_un_objeto xs)) ⇒ elem (Right o) (Left a):xs ≡ (evaluando es_un_objeto (Left a) según la definición de es_un_objeto)

∀ a :: Personaje . ∀ o :: Objeto . elem o (map (\(Right x)-> x) (if False then (Left a):(filter es_un_objeto xs) else (filter es_un_objeto xs)) ⇒ elem (Right o) (Left a):xs ≡

∀ a :: Personaje . ∀ o :: Objeto . elem o (map (\(Right x)-> x) (filter es_un_objeto xs) ⇒ elem (Right o) (Left a):xs ≡ (reemplazo sintáctico por la definición de objetos_en xs)

∀ a :: Personaje . ∀ o :: Objeto . elem o (objetos_en xs) ⇒ elem (Right o) (Left a):xs ≡ (por la definición de elem)

∀ a :: Personaje . ∀ o :: Objeto . elem o (objetos_en xs) ⇒ (Right o) = (Left a) || elem (Right o) xs ≡ (evaluando (Right o) = (Left a))

∀ a :: Personaje . ∀ o :: Objeto . elem o (objetos_en xs) ⇒ False || elem (Right o) xs ≡

∀ o :: Objeto . elem o (objetos_en xs) ⇒ elem (Right o) xs

VÁLIDO POR HIPÓTESIS INDUCTIVA

Subcaso Right

∀ b :: Objeto . ∀ o :: Objeto . elem o (objetos_en (Right b):xs) ⇒ elem (Right o) (Right b):xs ≡ (por definición de objetos_en)

∀ b :: Objeto . ∀ o :: Objeto . elem o (map (\(Right x)-> x) (filter es_un_objeto (Right b):xs)) ⇒ elem (Right o) (Right b):xs ≡ (por definición de filter)

∀ b :: Objeto . ∀ o :: Objeto . elem o (map (\(Right x)-> x) (if es_un_objeto (Right b) then (Right b):(filter es_un_objeto xs) else filter es_un_objeto xs)) ⇒ elem (Right o) (Right b):xs ≡ (evaluando es_un_objeto (Right b) según la definición de es_un_objeto)

∀ b :: Objeto . ∀ o :: Objeto . elem o (map (\(Right x)-> x) (if True then (Right b):(filter es_un_objeto xs) else filter es_un_objeto xs)) ⇒ elem (Right o) (Right b):xs ≡

∀ b :: Objeto . ∀ o :: Objeto . elem o (map (\(Right x)-> x) (Right b):(filter es_un_objeto xs)) ⇒ elem (Right o) (Right b):xs ≡ (por definición de map)

∀ b :: Objeto . ∀ o :: Objeto . elem o ((\(Right x)-> x) (Right b):(map (\(Right x)->x)(filter es_un_objeto xs))) ⇒ elem (Right o) (Right b):xs ≡ (evaluando ((\(Right x)-> x) (Right x) y reemplazo sintáctico por la definición de objetos_en xs)

∀ b :: Objeto . ∀ o :: Objeto . elem o (b:(objetos_en xs)) ⇒ elem (Right o) (Right b):xs ≡ (por definición de elem)

∀ b :: Objeto . ∀ o :: Objeto . o=b || elem o (objetos_en xs) ⇒ (Right o)=(Right b) || elem (Right o) xs

Veamos por casos:

- o=b || elem o (objetos_en xs) = False

  La implicación resulta verdadera, pues False ⇒ P es True ∀P

- o=b || elem o (objetos_en xs) = True
  - o=b = True
    o=b ⇒ (Right o)=(Right b) Luego, la implicación resulta verdadera
  - elem o (objetos_en xs) = True

elem o (objetos_en xs) ⇒ elem (Right o) xs VÁLIDO POR HIPÓTESIS INDUCTIVA, Luego la implicación resulta verdadera ■

QED.