A Clear Picture of Lens Laws

Sebastian Fischer
2015
Scale width only

Scaled to same width

Scale nicely
scaleWidth : Pic → Width → Pic
scaleNicely : Pic → Width → Pic

width : Pic → Width

(in Haskell-like notation)
Bidirectional Correspondence

\[
\text{width}(\text{scaleWidth } p \ w) = w \\
\text{scaleWidth } p (\text{width } p) = p
\]

\[
\text{width}(\text{scaleNicely } p \ w) = w \\
\text{scaleNicely } p (\text{width } p) = p
\]
Generally

\[
\text{put : Source } \rightarrow \text{ View } \rightarrow \text{ Source} \\
\text{get : Source } \rightarrow \text{ View}
\]

**PutGet:** \[\text{get (put s v)} = v\]

**GetPut:** \[\text{put s (get s)} = s\]

\[\rightarrow \text{ well-behaved lens}\]
As not every combination of put and get functions is well-behaved, which part is redundant?

(put is not!)
get is redundant

Let \((\text{put}, \text{get}_1)\) and \((\text{put}, \text{get}_2)\) be well-behaved.

Then for all sources \(s\):

\[
\text{get}_1 s = \text{get}_1 (\text{put} s (\text{get}_2 s)) = (\text{GetPut}_2)
\]

\[
= \text{get}_2 s = (\text{PutGet}_1)
\]
So, in a well-behaved lens, `get` is determined by `put`.

But, when does a corresponding `get` exist for an arbitrary given function `put : Source \to View \to Source`?

(such that they form a well-behaved lens)
Looking at put alone

scaleNot : Pic → width → Pic
scaleNot p w = p

Is there a corresponding get function?
Put Injectivity

scaleWidth \( p : \text{Width} \rightarrow \text{Pic} \)

gives different results for different widths

scaleNicely \( p : \text{Width} \rightarrow \text{Pic} \)

gives different results for different widths

scaleNot \( p : \text{Width} \rightarrow \text{Pic} \)

gives same result for different widths
Put injectivity is implied by lens laws (proof omitted).

\[ \Rightarrow \text{It is a necessary condition on put functions in well-behaved lenses} \]
No corresponding get for

scaleNot : Pic → width → Pic
scaleNot p w = p

because injectivity does not hold.
Looking at put alone

$\text{scaleBy : Pic \rightarrow Factor \rightarrow Pic}$

$\text{scaleBy } p \times =$

$\text{scaleNicely } p \ (x \cdot \text{width } p)$

Is there a corresponding get function?
Put Twice

$\text{scaleWidth} \ (\text{scaleWidth} \ p \ w) \ w$
$= \text{scaleWidth} \ p \ w$

$\text{scaleNicely} \ (\text{scaleNicely} \ p \ w) \ w$
$= \text{scaleNicely} \ p \ w$

but

$\text{scaleBy} \ (\text{scaleBy} \ p \ 2) \ 2$
$= \text{scaleBy} \ p \ 4$
$\neq \text{scaleBy} \ p \ 2$
PutTwice is implied by lens laws:

\[
\begin{align*}
&\text{put} \ (\text{put} \ s \ v) \ v \\
= &\text{put} \ (\text{put} \ s \ v) \ (\text{get} \ (\text{put} \ s \ v)) \ (\text{PutGet}) \\
= &\text{put} \ s \ v \ (\text{GetPut}) \\
\end{align*}
\]

→ It is a necessary condition on \text{put} functions in well-behaved lenses.
No corresponding get for

scaleBy: Pic -> Factors -> Pic

because Put Twice does not hold.

(scaleBy satisfies Put Injectivity and
scale Not satisfies Put Twice)
Looking at put alone

\[ \text{scaleBlank : Pic} \rightarrow \text{Width} \rightarrow \text{Pic} \]

\[ \text{scaleNiceLy someBlankPic w} \]

Is there a corresponding get function?
Put Surjectivity

every picture is result of scale width
(by scaling itself with its own width)

every picture is result of scale nicely
(by scaling itself with its own width)

not every picture is result of scale blank
(because every result is blank)
Surjectivity is implied by lens laws (proof omitted).

It is a necessary condition on put functions in well-behaved lenses.
No corresponding get for scaleBlank: Pic \rightarrow \text{Width} \rightarrow \text{Pic}

because \text{PutSurjectivity does not hold.}

(scaleNot and scaleBy satisfy PutSurjectivity, scaleBlank satisfies PutInjectivity and PutTwice)
PutInjectivity, PutTwice, PutSurjectivity are necessary conditions on put. They are also sufficient for existence of corresponding get that forms a well-behaved lens with put. (proof omitted)

(remembers: get is unique)
Put Get

View Determination

Get Put

Source Stability

Put Twice

Put Injectivity

Put Surjectivity

0 => existence of get, so 0 E => 0