### 0.0.1 Update At

The primitive *updateAt* performs a replacement at some depth within a Collection without changing any other values in the source Collection.

$$\mathit{UpdateAt}[\mathit{Collection}, V, \mathit{Collection}]$$

$$\begin{array}{l}
\mathit{AtDepth} \\
\mathit{coll}?\,, \mathit{coll}!\,, \mathit{indices}? : \mathit{Collection} \\
v? : V \\
\mathit{updateAt}\,\_ : \mathit{Collection} \times V \times \mathit{Collection} \rightarrowtail\!\!\!\rightarrow \mathit{Collection} \\
\hline
\mathit{updateAt} = \langle \mathit{atDepth}\,\_, \mathit{update}\,\_, \langle \mathit{atDepth}\,\_, \mathit{update}\,\_\rangle^{\#\,\mathit{indices}?-1} \rangle \\
\mathit{coll}! = \mathit{updateAt}(\mathit{coll}?\,, v?\,, \mathit{indices}?\,) \bullet \\
\quad \mathit{let} \quad \mathit{coll} == \mathit{update}(\mathit{atDepth}(\mathit{coll}?\,, \mathit{indices}? \lhd \mathit{indices}?_j\,), v?\,, \mathit{indices}?_j\,) \bullet \\
\qquad \forall n : i \mathrel{..} j-1 \bullet j = \mathit{first}(\mathit{last}(\mathit{indices}?\,)) \mid \exists\, c_n \bullet \\
\qquad\quad \mathit{let} \quad c_i == \mathit{atDepth}(\mathit{coll}?\,, (\mathit{indices}? \upharpoonright i)) \Rightarrow \mathit{atIndex}(\mathit{coll}?\,, \mathit{indices}?_i\,) \\
\qquad\qquad\quad c_n == \mathit{atDepth}(c_{n-1}, (\mathit{indices}? \upharpoonright n)) \\
\qquad\qquad\quad c_{j-1} == \mathit{atDepth}(c_n, (\mathit{indices}? \upharpoonright j-1)) \iff n = j-2 \\
\qquad\qquad\quad c_j == \mathit{update}(c_{j-1}, v?\,, (\mathit{indices}? \upharpoonright j)) \Rightarrow c_j = \mathit{coll} = \mathit{coll}!_j \\
\qquad\qquad\quad \mathit{coll}!_{j-2} == \mathit{update}(c_{j-2}, c_j, \mathit{indices}?_{j-1}\,) \\
\qquad\qquad\quad \mathit{coll}!_n == \mathit{update}(c_{n-1}, \mathit{coll}!_n\,, \mathit{indices}?_n\,) \\
\qquad\qquad\quad \mathit{coll}!_i == \mathit{update}(c_i, \mathit{coll}!_n\,, \mathit{indices}?_n\,) \iff n = i+1 \\
\quad = \mathit{update}(\mathit{coll}?\,, \mathit{coll}!_i\,, \mathit{indices}?_i\,)
\end{array}$$

The following examples demonstrate the properties of *updateAt*

$$\begin{array}{ll}
X = \langle x_0, x_1, x_2 \rangle & \\
\quad x_0 = 0 & \\
\quad x_1 = \mathit{foo} & \\
\quad x_2 = \langle a, b, c \rangle & \\
\mathit{updateAt}(X, \langle 1, 3 \rangle, z) = \langle x_0, \mathit{fooz}, x_2 \rangle \Rightarrow \mathit{foo} = \langle f, o, o \rangle & \\
\mathit{updateAt}(X, \langle 1, 0 \rangle, z) = \langle x_0, \mathit{zoo}, x_2 \rangle \Rightarrow \mathit{zoo} = \langle z, o, o \rangle & \\
\mathit{updateAt}(X, \langle 0 \rangle, 5) = \langle 5, x_1, x_2 \rangle & \\
\mathit{updateAt}(X, \langle 0, 0 \rangle, 5) = \langle \langle 5 \rangle, x_1, x_2 \rangle & [\text{5 in seq}] \\
\mathit{updateAt}(X, \langle 0, 1 \rangle, 5) = \langle \langle 0, 5 \rangle, x_1, x_2 \rangle & [\text{5 and 0 in seq}] \\
\mathit{updateAt}(X, \langle 2, 0 \rangle, d) = \langle x_0, x_1, \langle d, b, c \rangle \rangle & \\
\mathit{updateAt}(X, \langle 2 \rangle, d) = \langle x_0, x_1, d \rangle &
\end{array}$$

Which indicates that if navigation would result in stepping into a non-nested Value, an empty sequence is created and then populated with the non-nested Value so navigation can continue.

$$\begin{array}{l}
X! = \mathit{updateAt}(X, \langle 0, 0 \rangle, 5) \mid i \mapsto 0 \wedge j \mapsto 0 \wedge i \neq j \bullet \\
\quad \mathit{let} \quad X_i == \mathit{atIndex}(X, i) = 0 \\
\qquad\quad X_{i+1} == \mathit{atIndex}(X_i, j) = \langle\rangle
\end{array}$$

$$\begin{aligned}
&\quad X_j == append(X_{i+1}, X_i, 0) = \langle 0 \rangle \qquad \text{[only item hence 0 index]}\\
&\quad X'_j == update(X_j, 5, j) = \langle 5 \rangle\\
&= update(X, X'_j, i)\\
&= \langle \langle 5 \rangle, x_1, x_2 \rangle
\end{aligned}$$

- The nesting which created $\langle 0 \rangle$ also specified that the first index of that sub Collection should have the value of 5 which is why $X! = \langle \langle 5 \rangle, x_1, x_2 \rangle$

$$\begin{aligned}
&X! = updateAt(X, \langle 0, 1 \rangle, 5) \mid i \mapsto 0 \wedge j \mapsto 1 \wedge i \neq j \bullet\\
&\quad let \quad X_i == atIndex(X, i) = 0\\
&\qquad X_{i+1} == atIndex(X_i, j) = \langle \rangle\\
&\qquad X_j == append(X_{i+1}, X_i, 0) = \langle 0 \rangle \qquad \text{[only item hence 0 index]}\\
&\qquad X'_j == update(X_j, 5, j) = \langle 0, 5 \rangle\\
&\quad = update(X, X'_j, i)\\
&\quad = \langle \langle 0, 5 \rangle, x_1, x_2 \rangle
\end{aligned}$$

- The nesting which created $\langle 0 \rangle$ now indicates that the second index of that sub Collection should have the value of 5 which is why $X! = \langle \langle 0, 5 \rangle, x_1, x_2 \rangle$

$$\begin{aligned}
&X! = updateAt(X, \langle 0 \rangle, 5) \mid i \mapsto 0 \bullet\\
&\qquad \# indices? = 1 \Rightarrow updateAt(X, \langle 0 \rangle, 5) \equiv update(X, i, 5) \bullet\\
&\quad = update(X, i, 5) = \langle 5, x_1, x_2 \rangle
\end{aligned}$$

- Here there is no further nesting which can't be performed so the update happens as if the Operation *update* was used.