

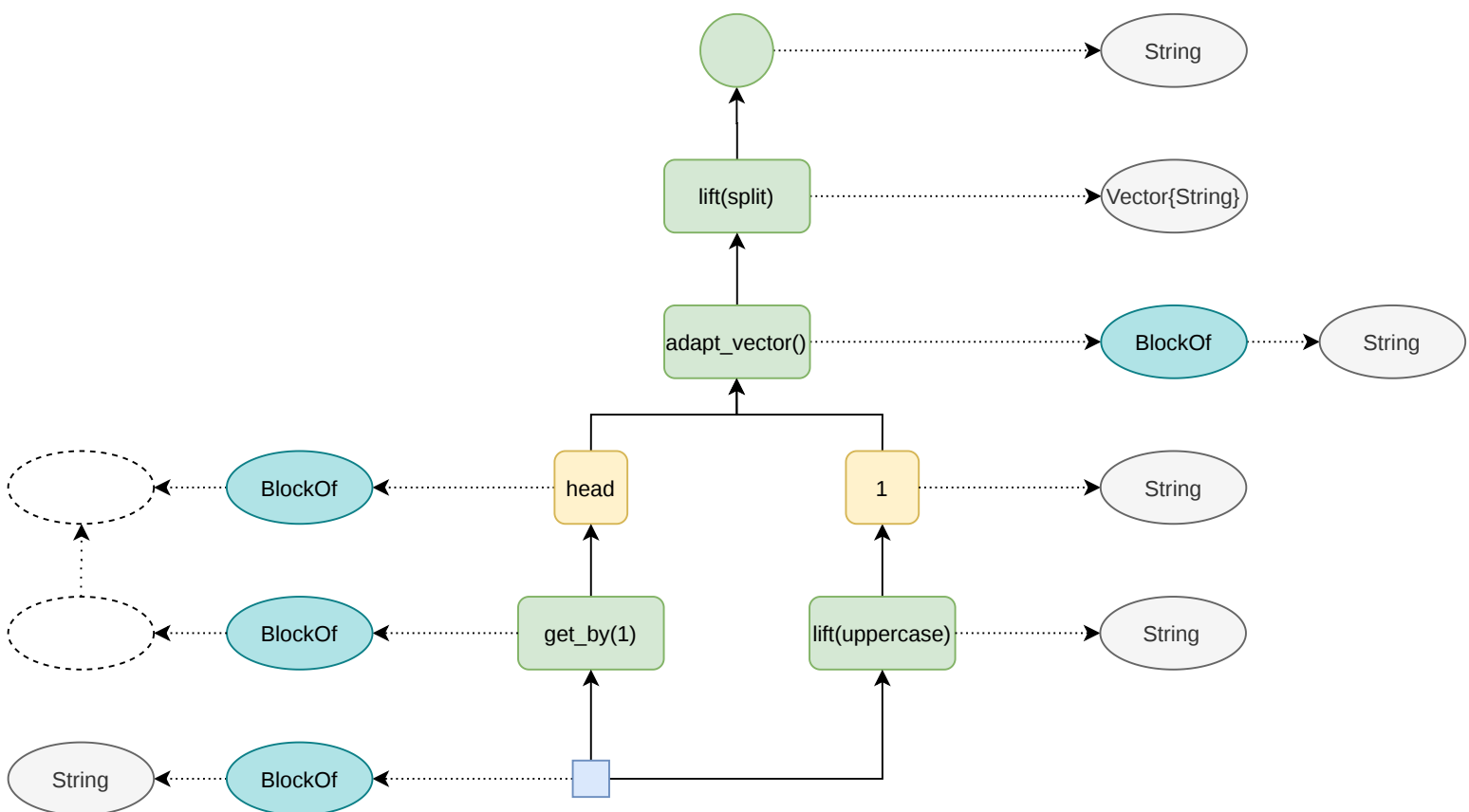
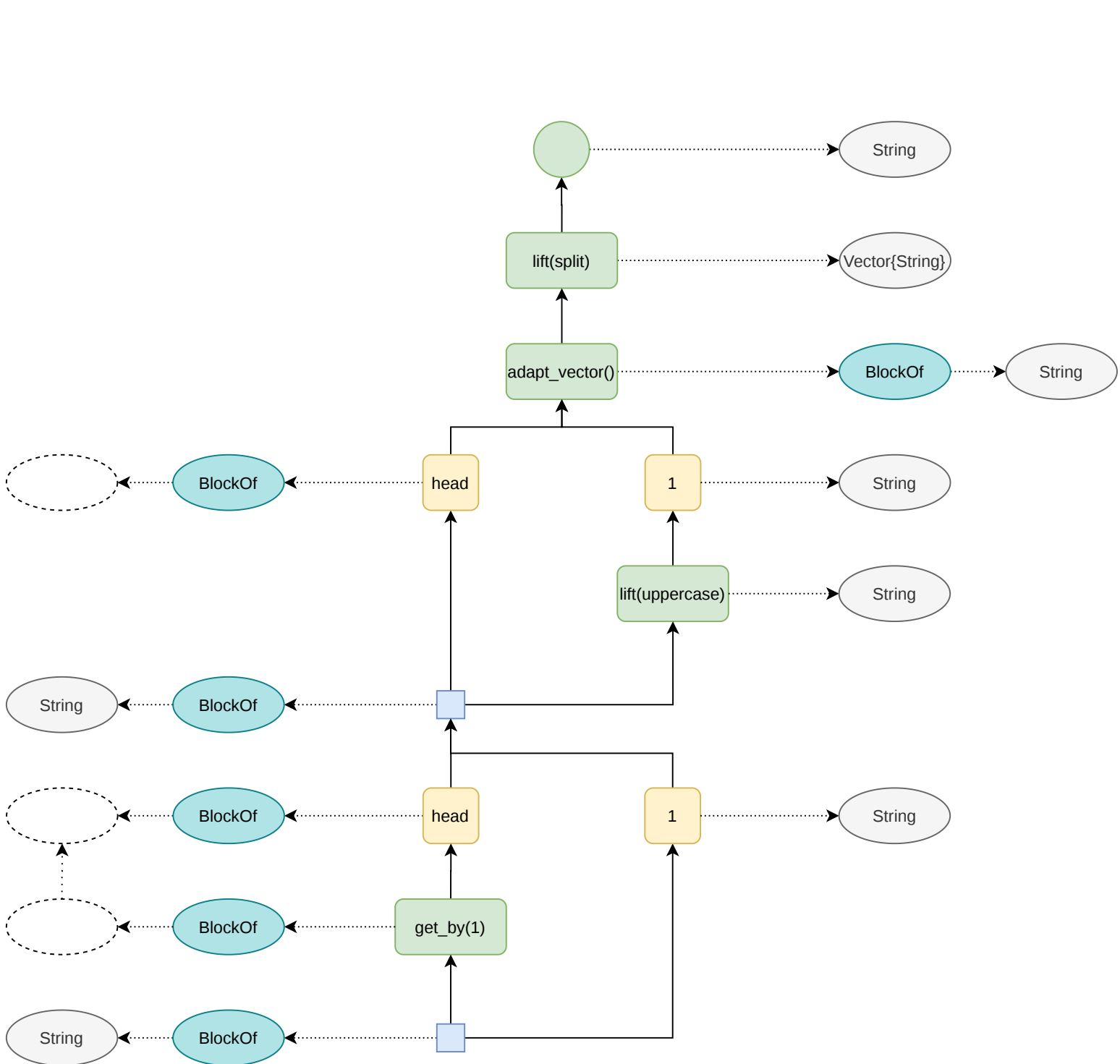
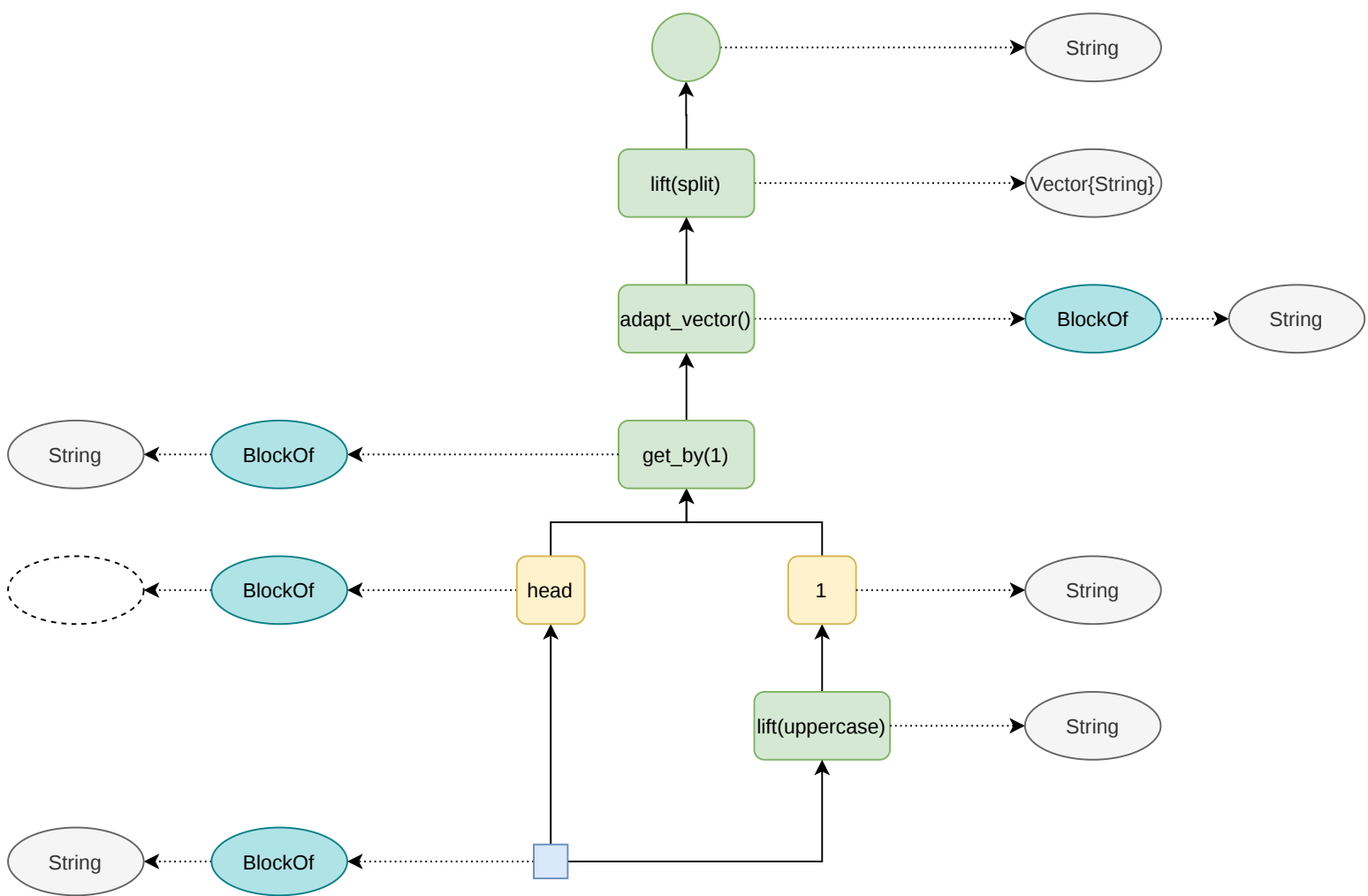
The diagram illustrates the transformation of a flat table into a hierarchical tree structure through four stages, connected by large gray arrows.

Stage 1: A flat table with two columns: an index column (1) and a value column ("Hello World").

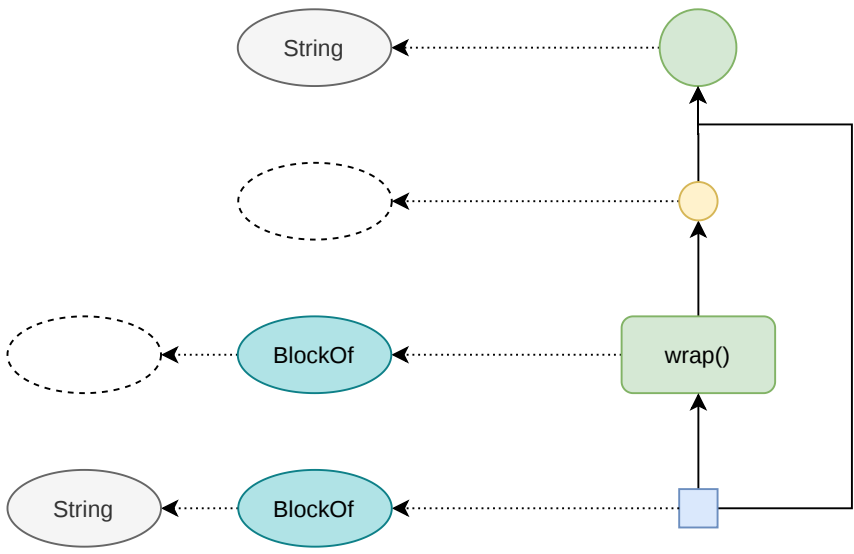
Stage 2: The table is transformed into a hierarchical structure. The root node is a table with one column (1) and one row (String["Hello", "World"]). This root node branches into two child nodes, each a table with two columns (1, 2) and two rows (1, 3) and (2, "World").

Stage 3: The hierarchical structure is further transformed. The root node branches into two child nodes, each a table with two columns (1, 2) and two rows (1, 3) and (2, "HELLO") and (2, "WORLD").

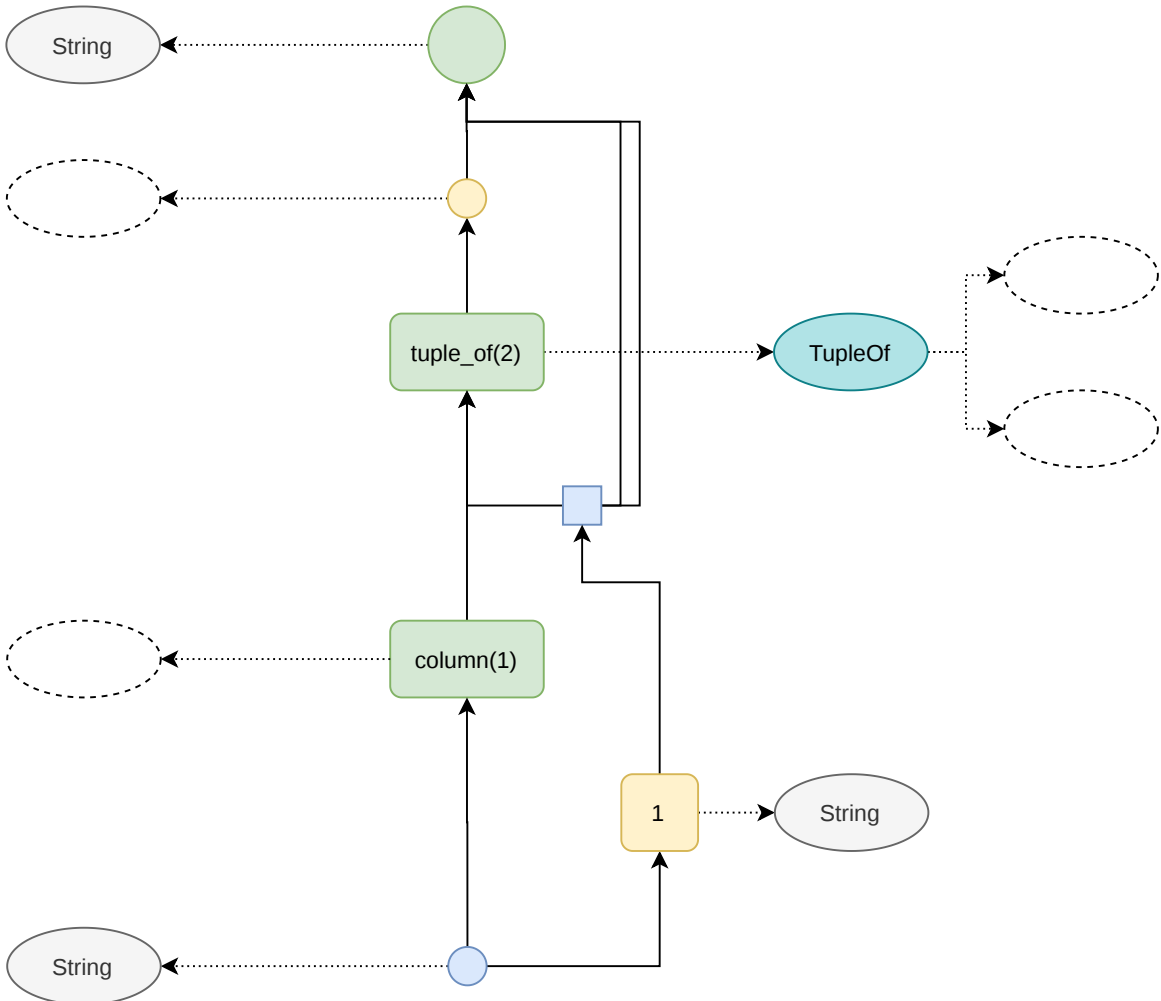
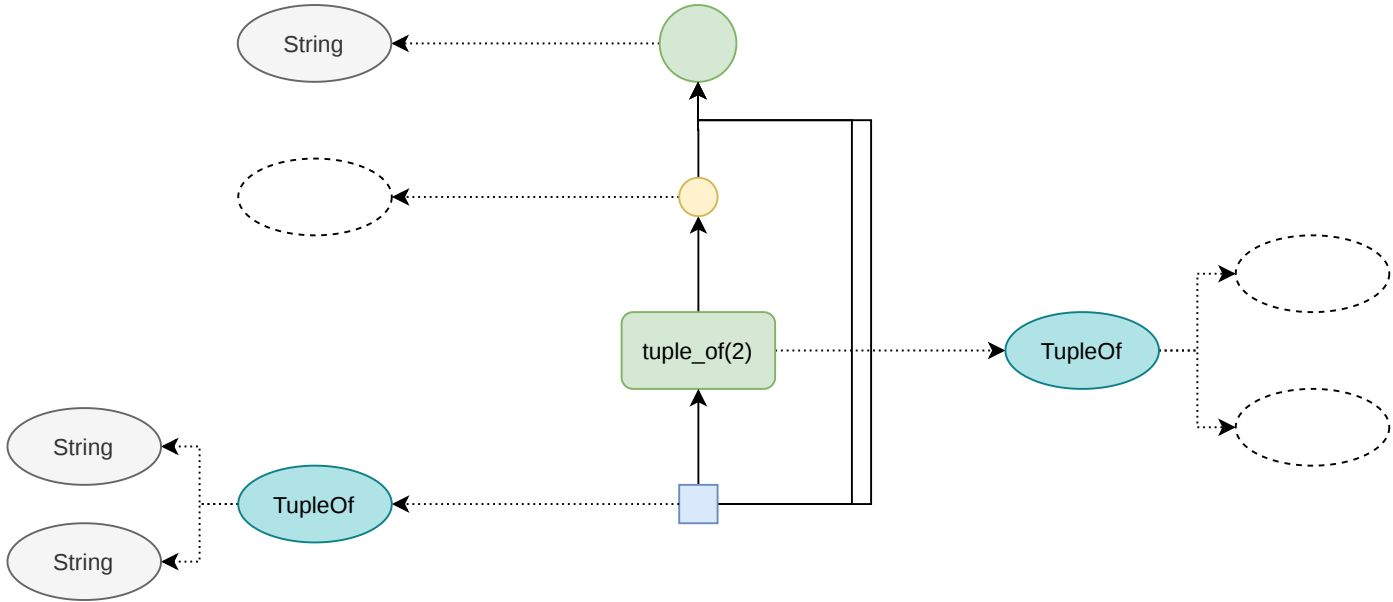
Stage 4: The hierarchical structure is further transformed. The root node branches into two child nodes, each a table with two columns (1, 2) and two rows (1, 3) and (2, 2) and (2, 2).



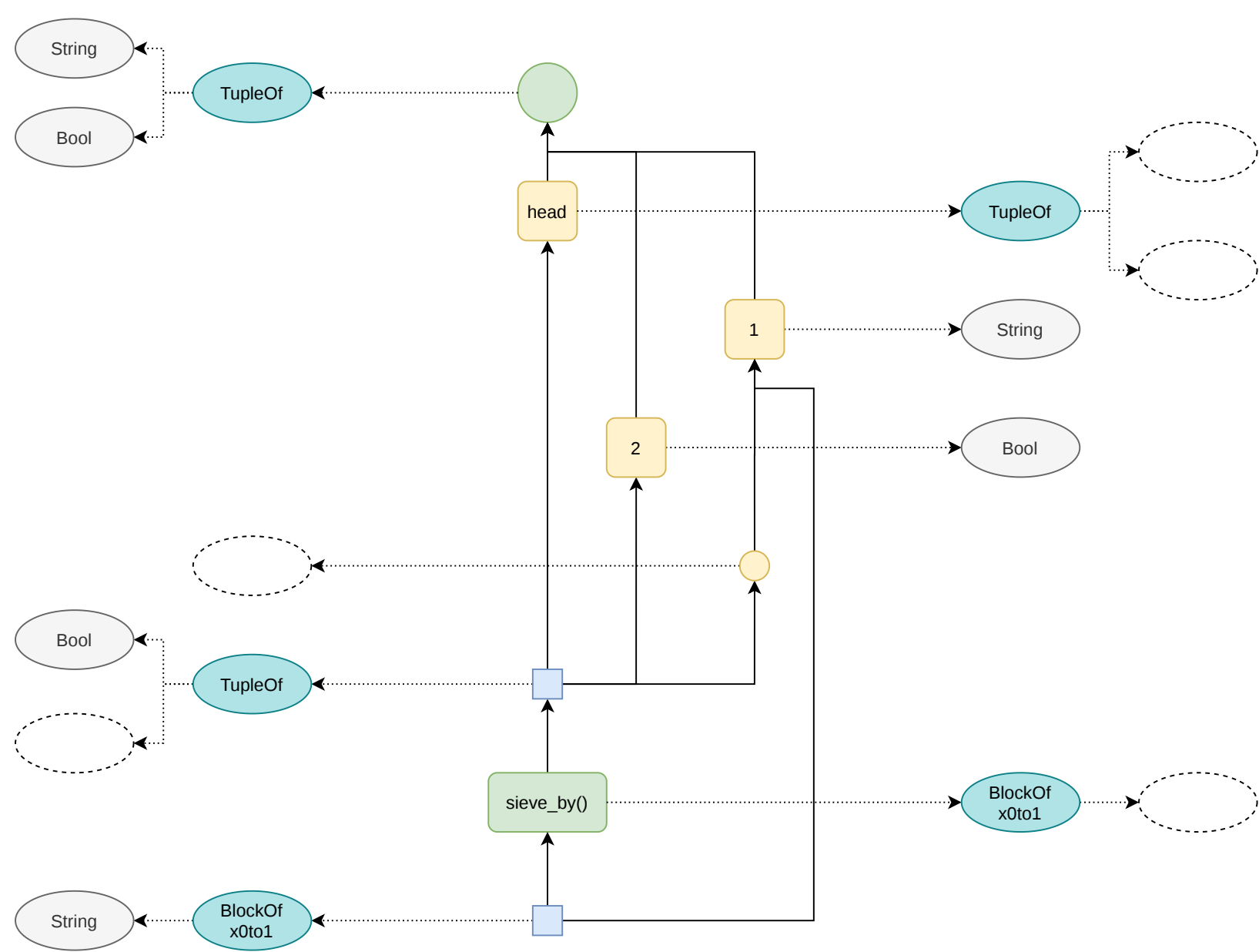
wrap0



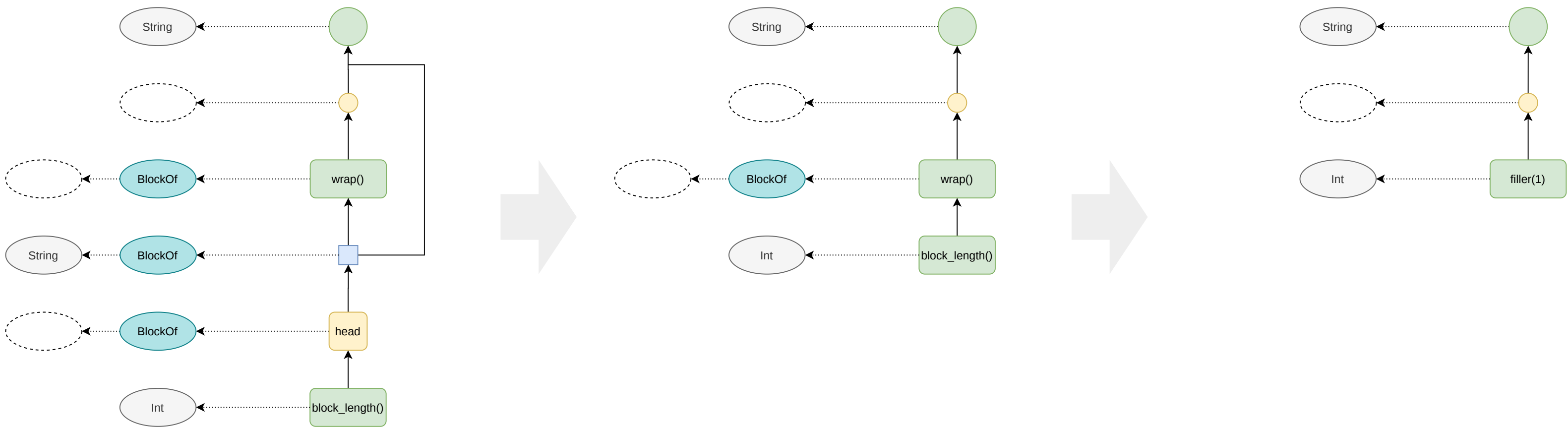
chain_of(tuple_of(2), column(1))



sieve_by()

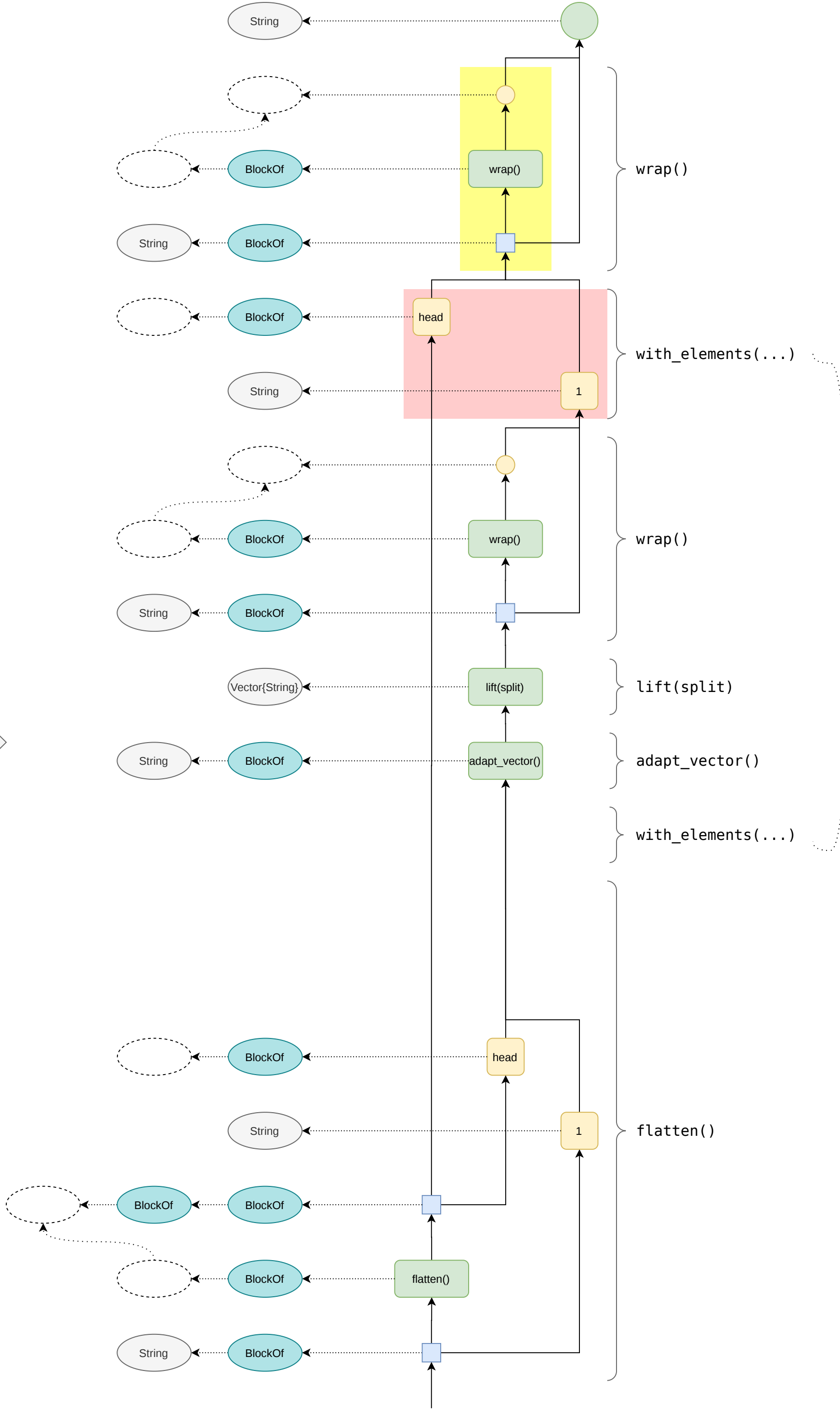
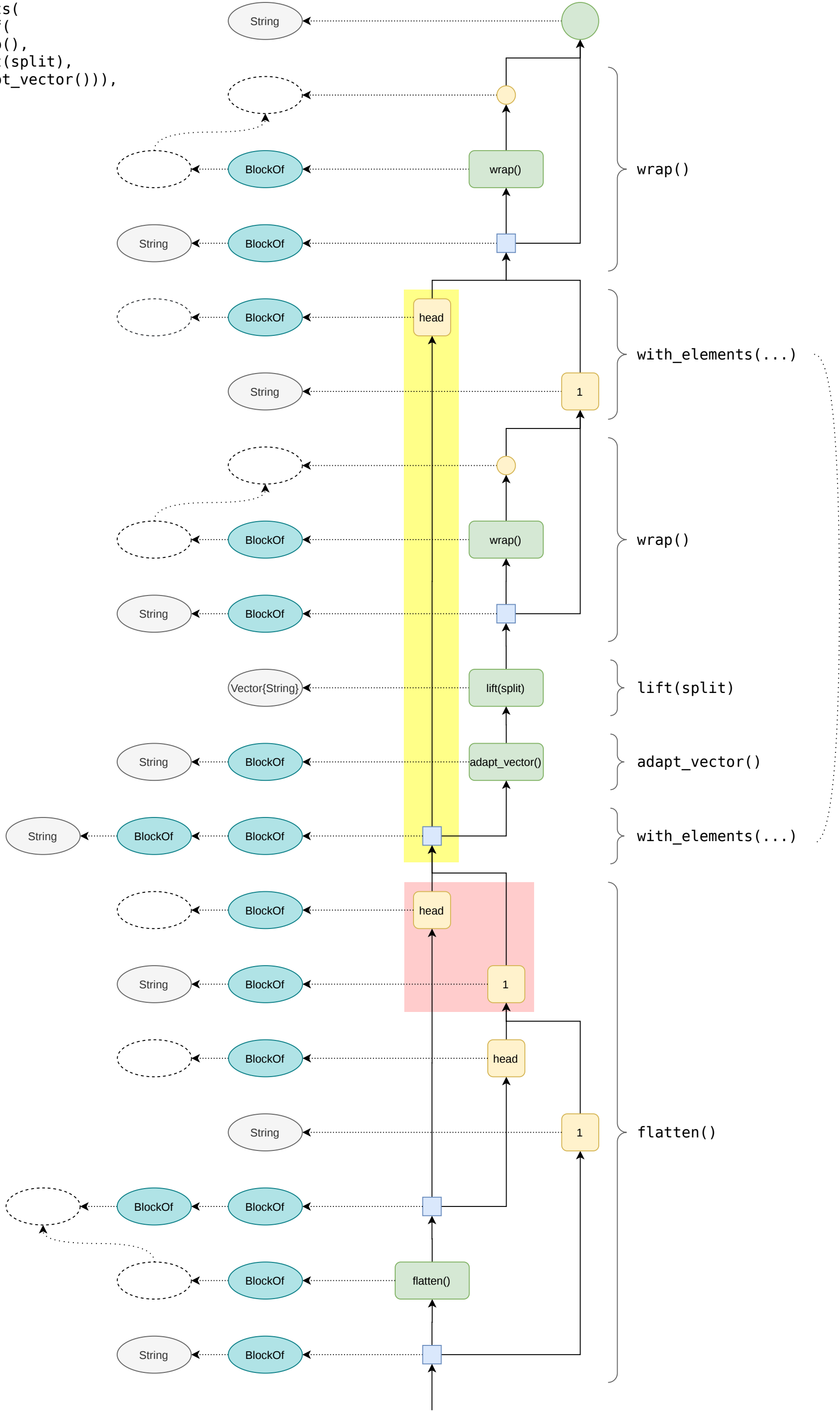


chain_of(wrap(), block_length())

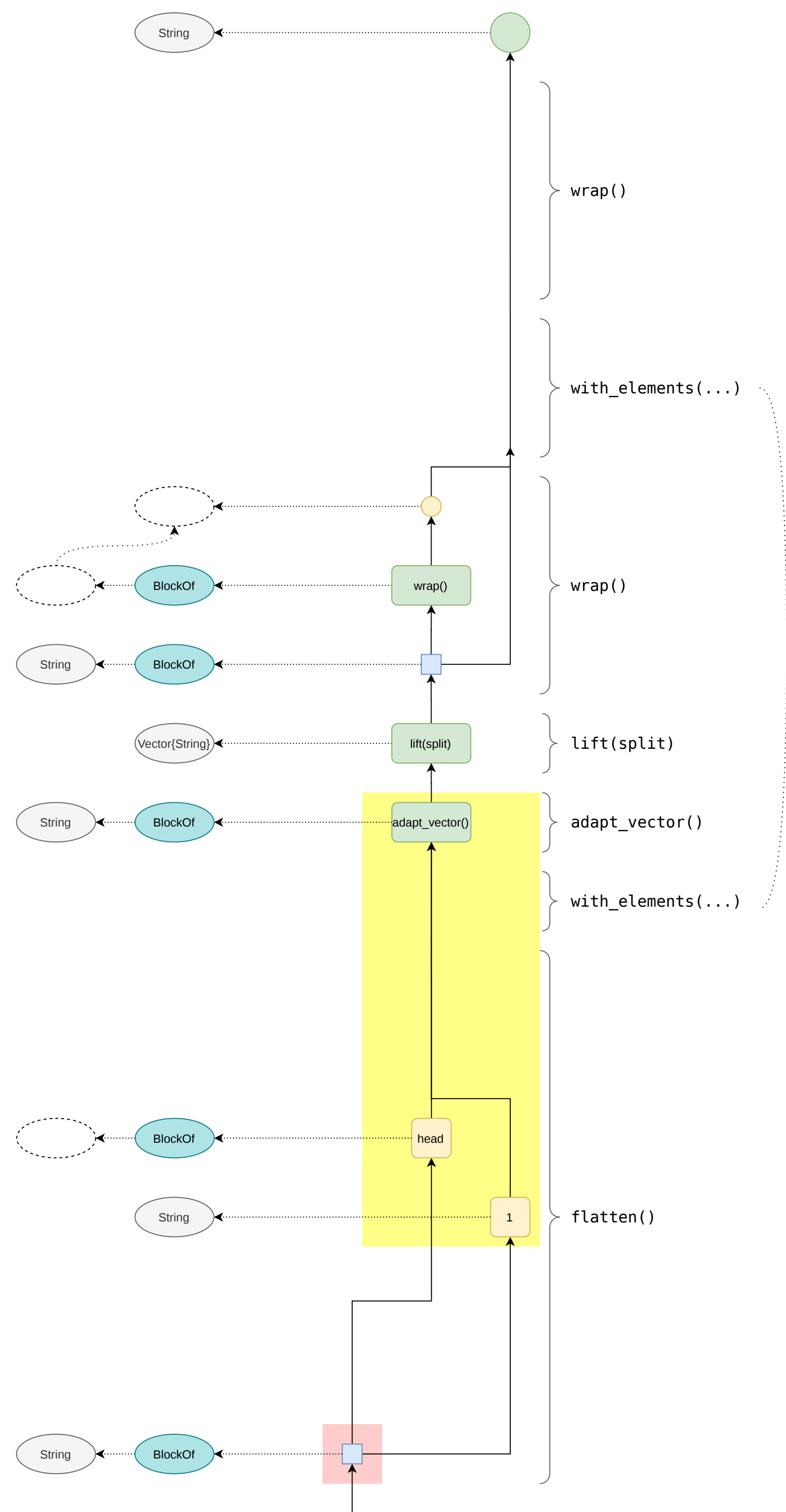
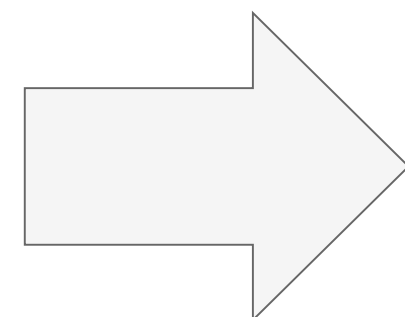


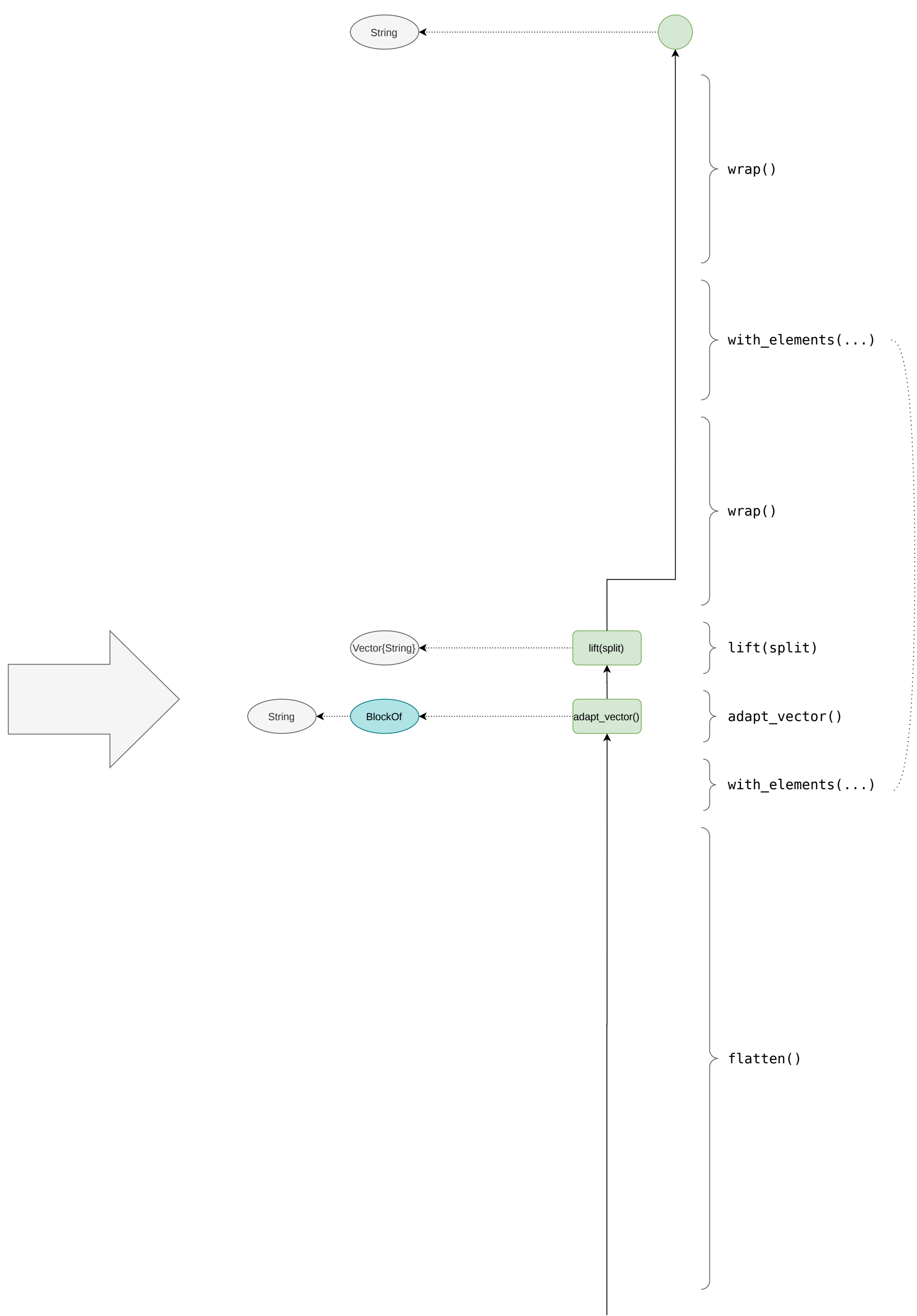
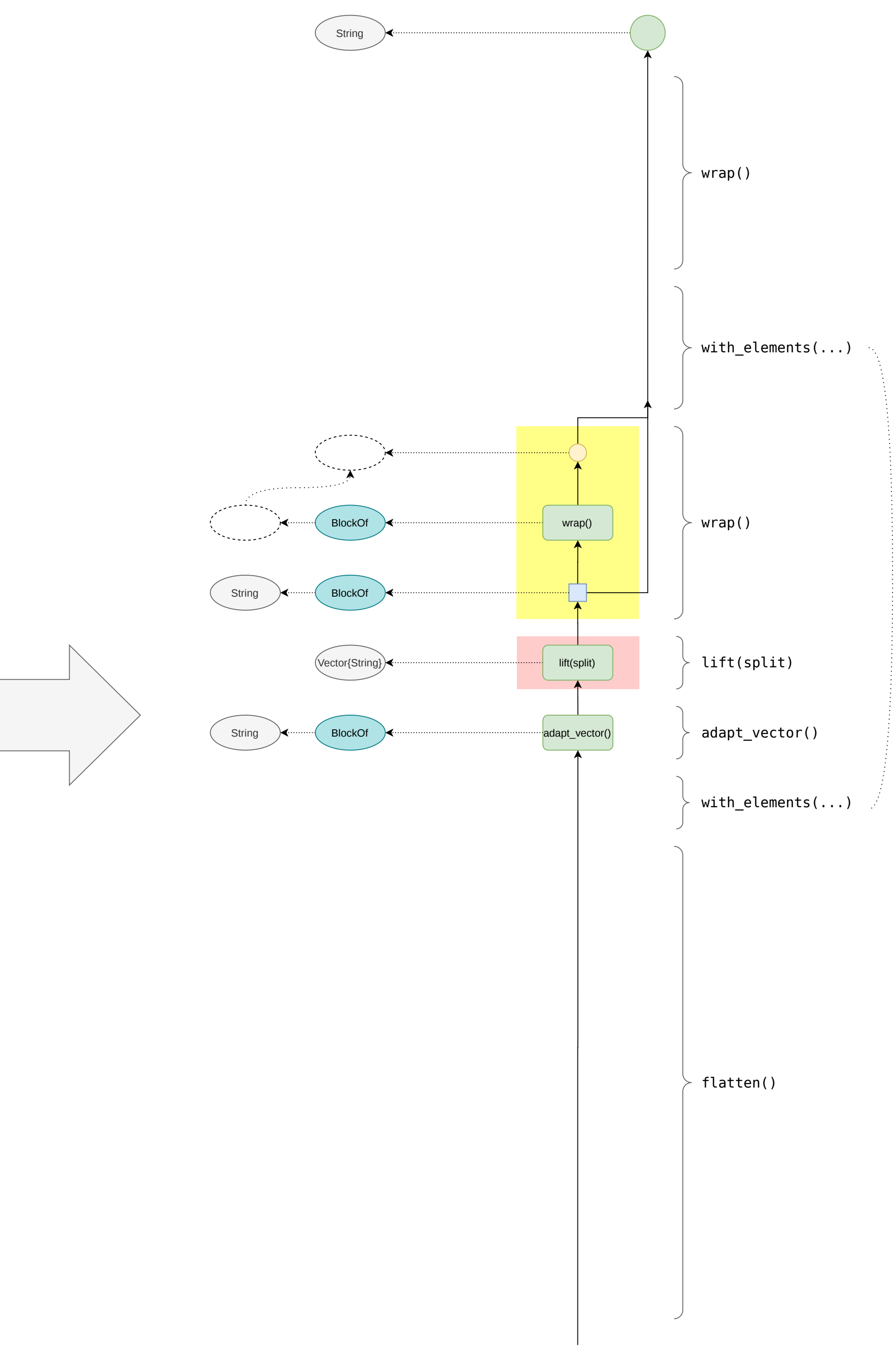
@query "Hello World" split(it)

```
chain_of(  
  wrap(),  
  with_elements(  
    chain_of(  
      wrap(),  
      lift(split),  
      adapt_vector()),  
    flatten()  
  )  
)
```



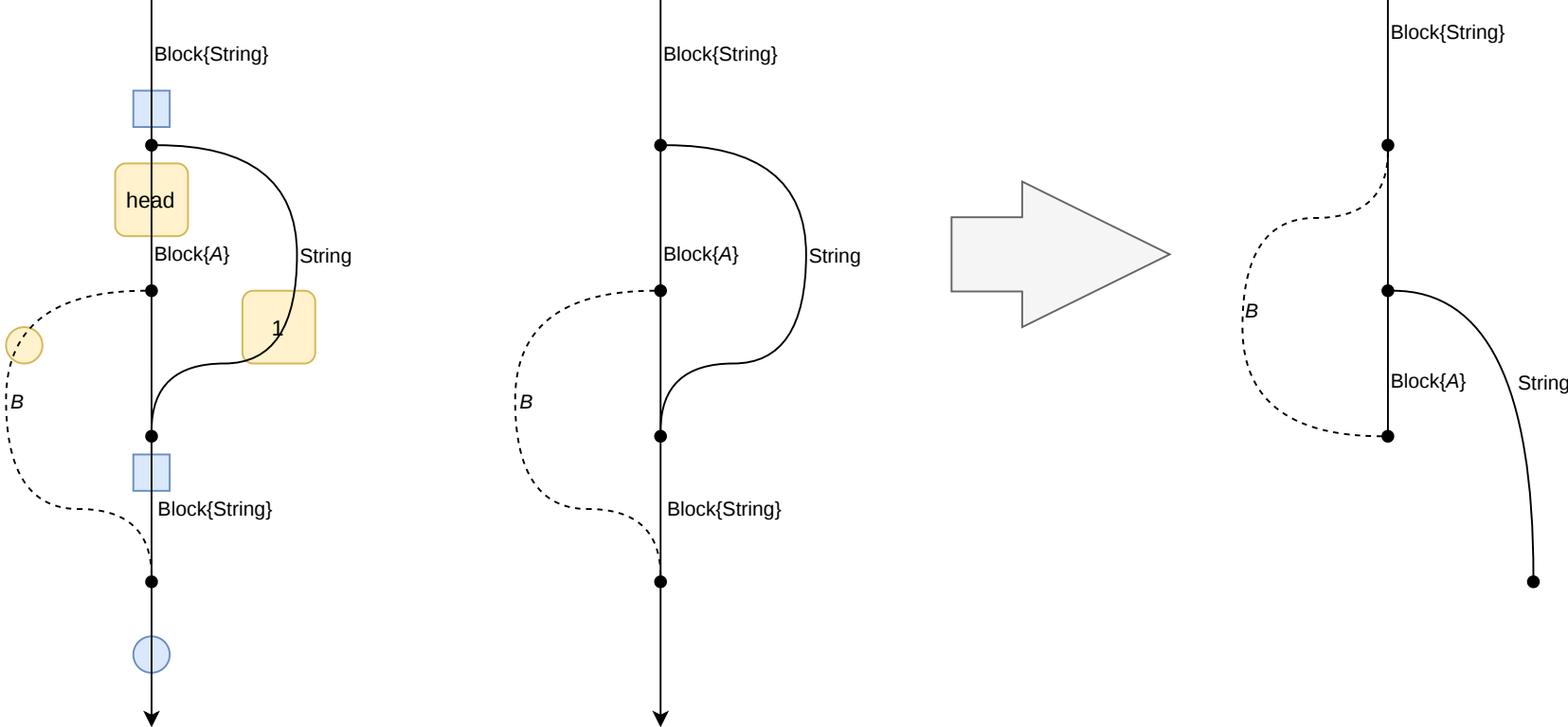
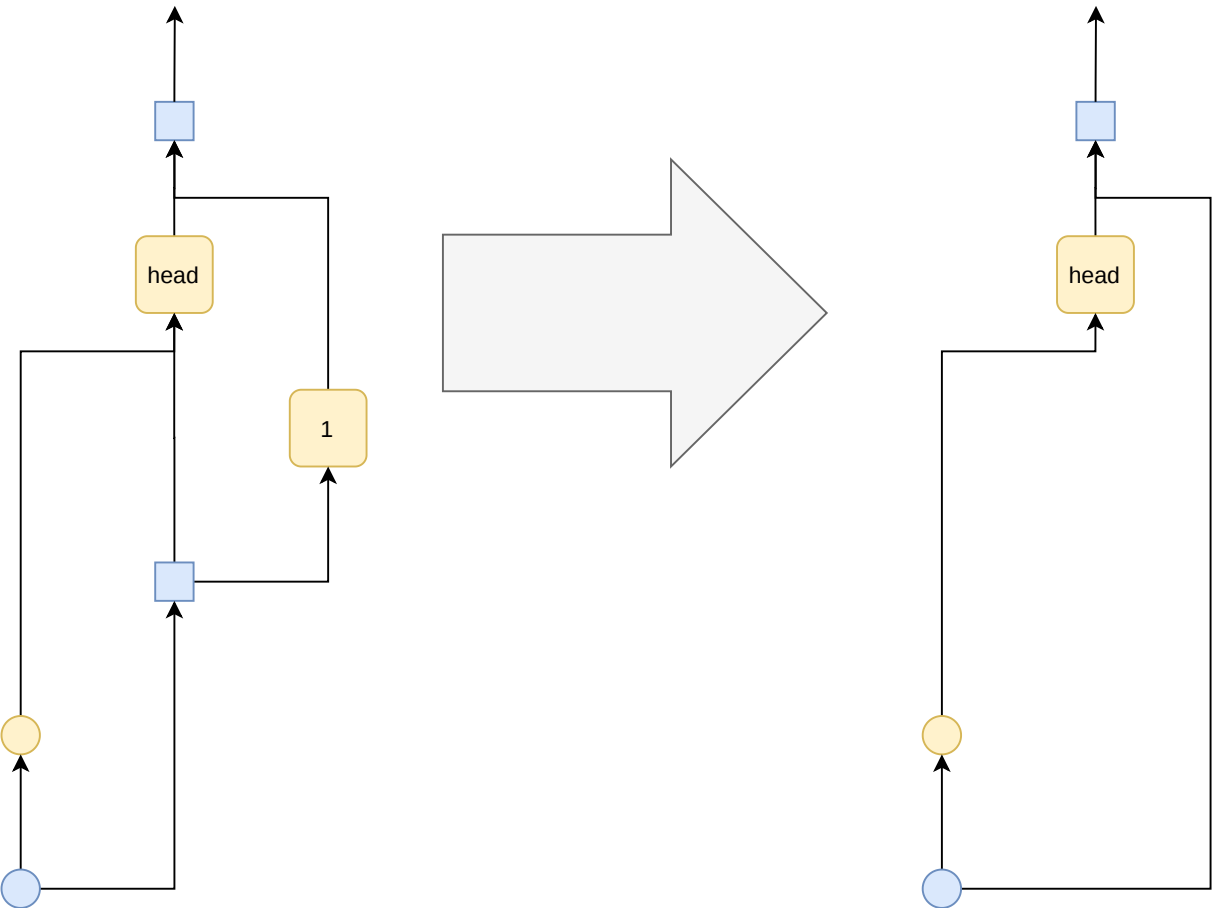
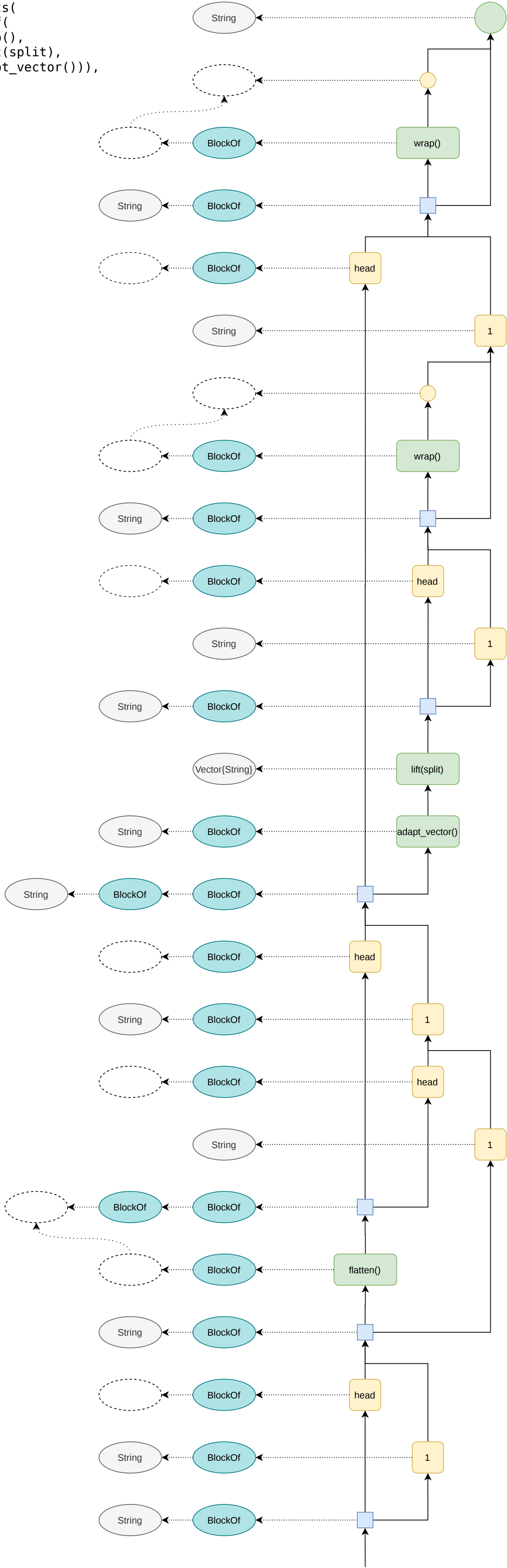
untrace(n::NodeRef, guard::NodeRef)::Tuple{Pipeline,Vector{NodeRef}}

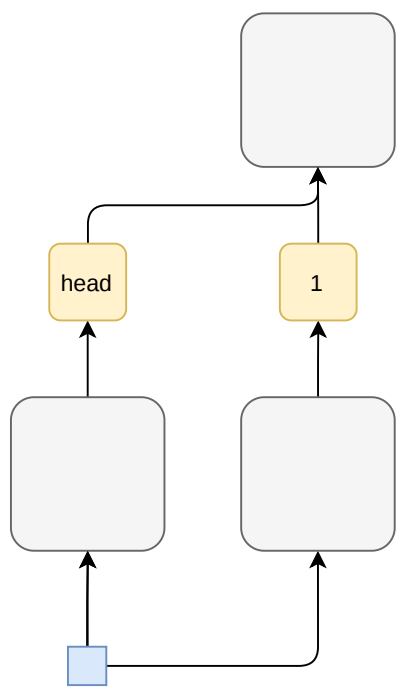
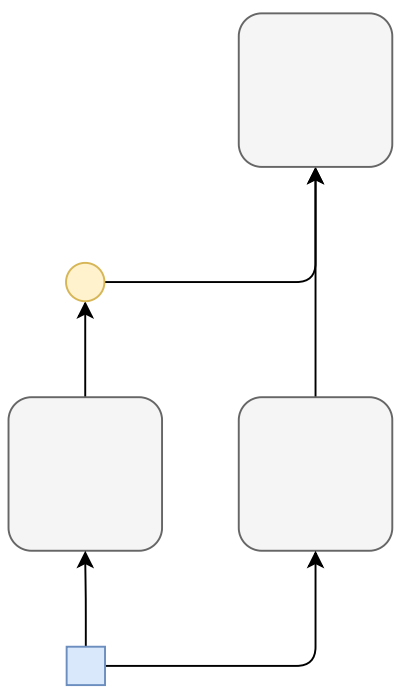
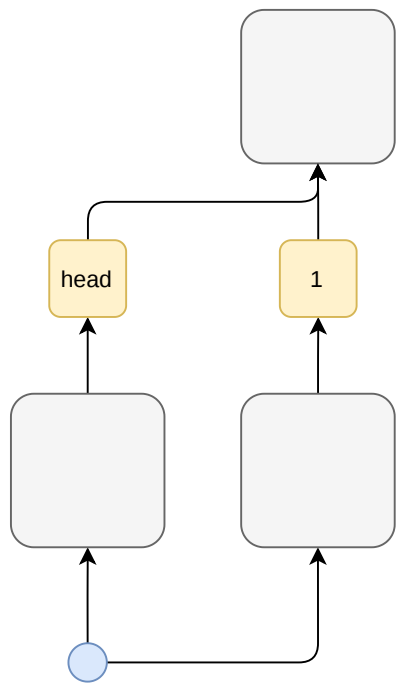




@query "Hello World" split(it)

```
chain_of(
  wrap(),
  with_elements(
    chain_of(
      wrap(),
      lift(split),
      adapt_vector()),
    flatten())
```



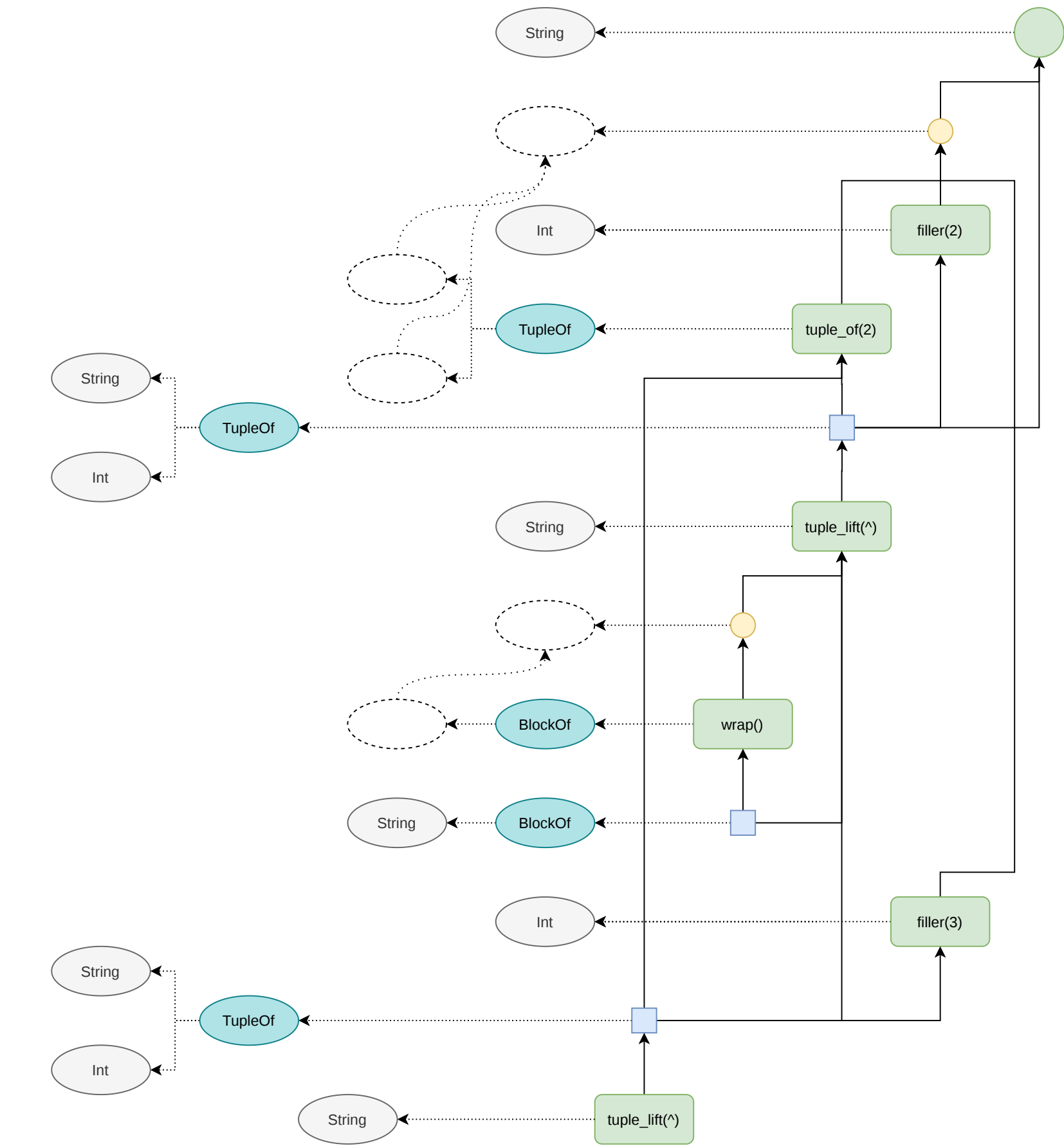




`untrace(n::NodeRef, guard::NodeRef)::Tuple{Pipeline,Vector{NodeRef}}`

`{it ^ 2, (it ^ 2) ^ 3}`

`chain_of(f, tuple_of(g, h)) => tuple_of(chain_of(f, g), chain_of(f, h))`



`untrace(n::NodeRef, guard::NodeRef)::Tuple{Pipeline,Vector{NodeRef}}`

@query "Hello World" split(it)

```
chain_of(
  wrap(),
  with_elements(wrap()),
  with_elements(lift(split)),
  with_elements(adapt_vector()),
  flatten())
```

