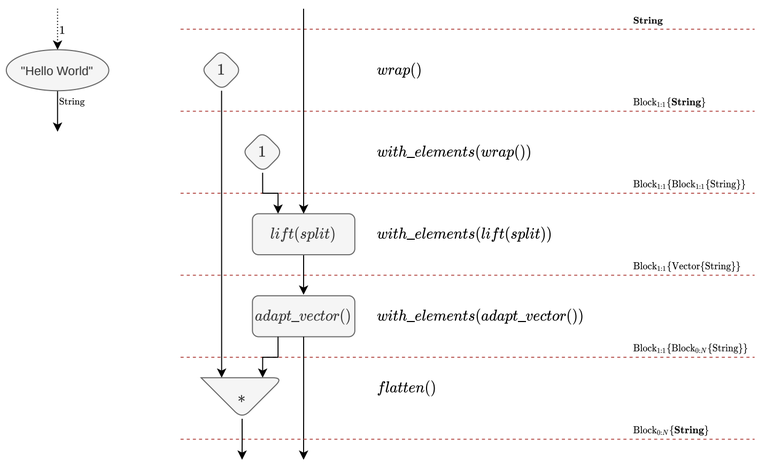
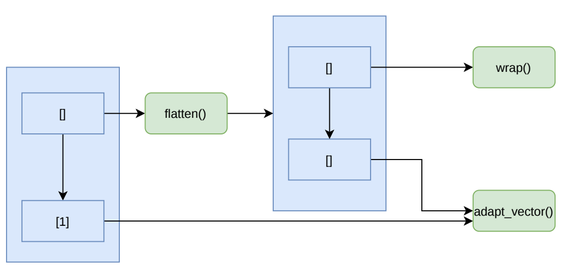
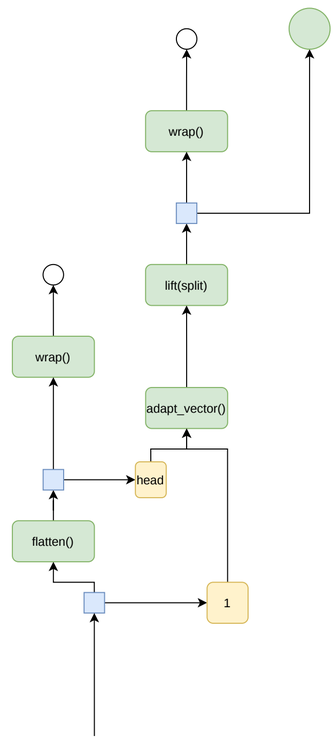
```
@query "Hello World" split(it)
```







flatten(): BlockOf{BlockOf{...}} -> BlockOf{...}

String

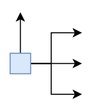
```
struct EvalNode
    p::Pipeline
    i::NodeRef
end
```

```
Nothing
```

```
struct SlotNode
    n::NodeRef
    k::Int
end
```

```
struct HeadNode
    n::NodeRef
end
```

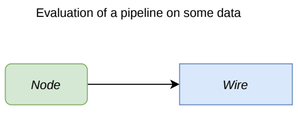
```
struct JoinNode
    head::NodeRef
    slots::Vector{NodeRef}
end
```



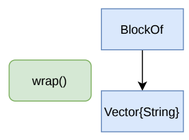
```
struct InputNode
end
```

```
mutable struct NodeRef
    node::Union{Nothing,Node}
end
```

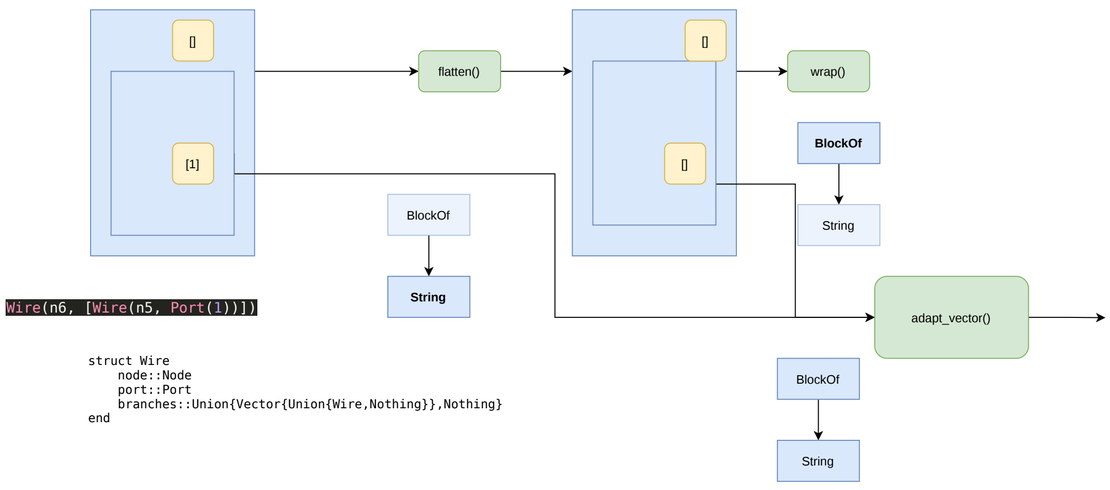
Evaluation of a pipeline on some data




```
struct Node
    p::Pipeline
    input::Union{Wire,Nothing}
end
```

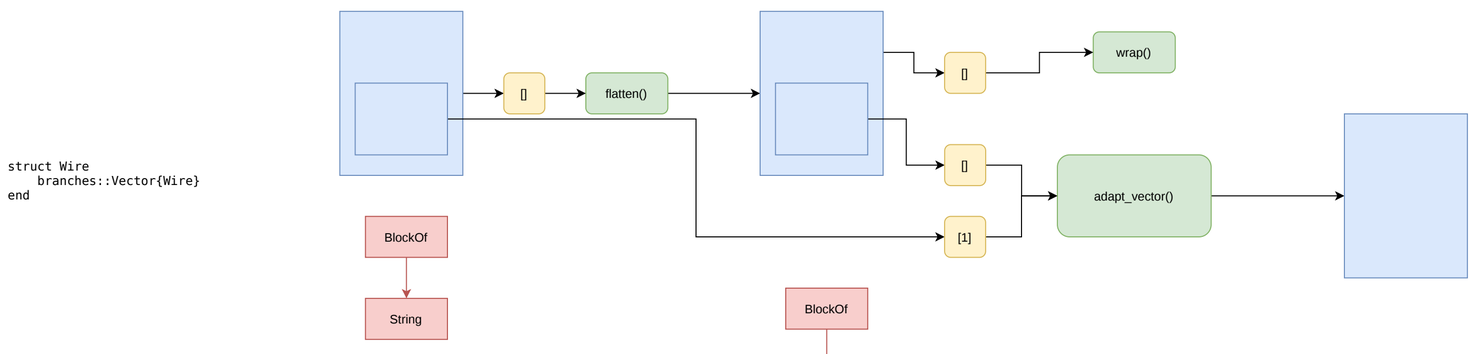


String



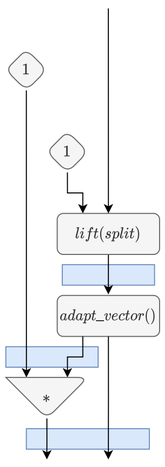

```
struct Wire
    node::Node
    port::Port
    branches::Union{Vector{Union{Wire,Nothing}},Nothing}
end
```

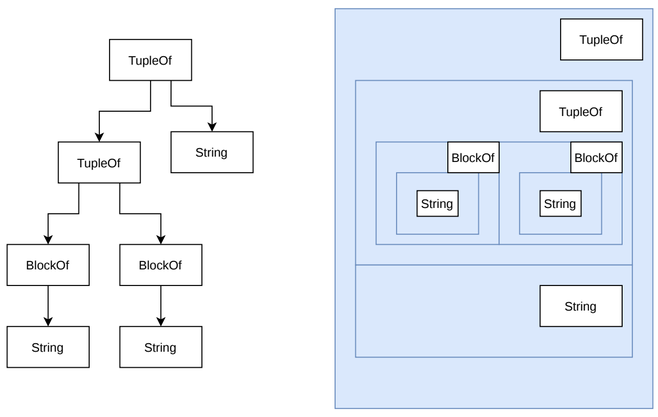
```
struct Node
    p::Pipeline
    input::Union{Wire,Nothing}
end
```



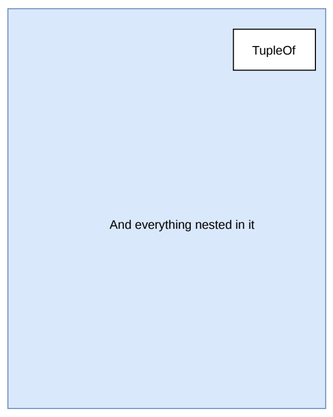

```
struct Wire
    branches::Vector{Wire}
end
```

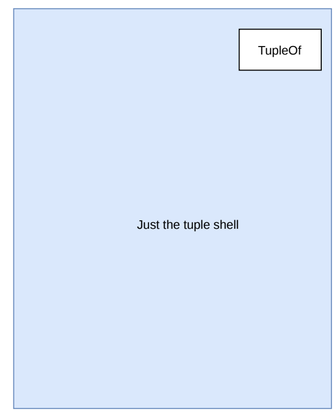
Vector{String}

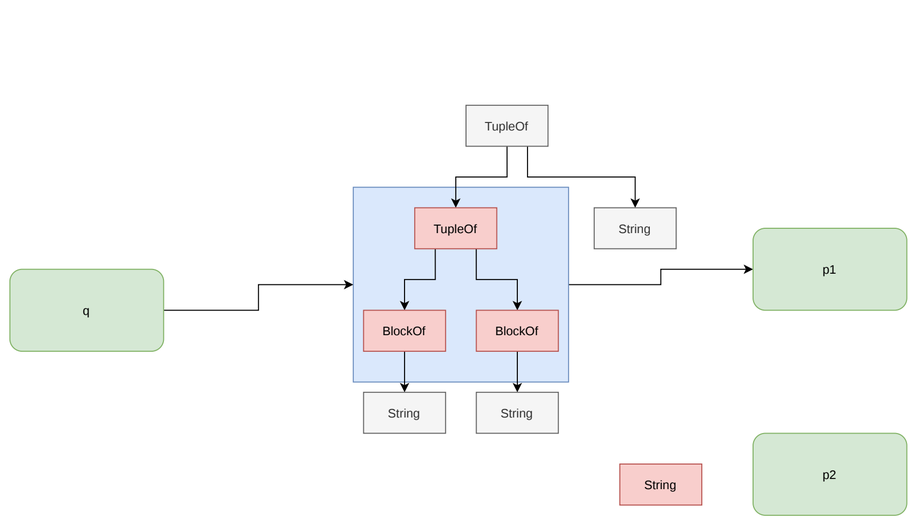


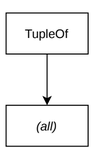


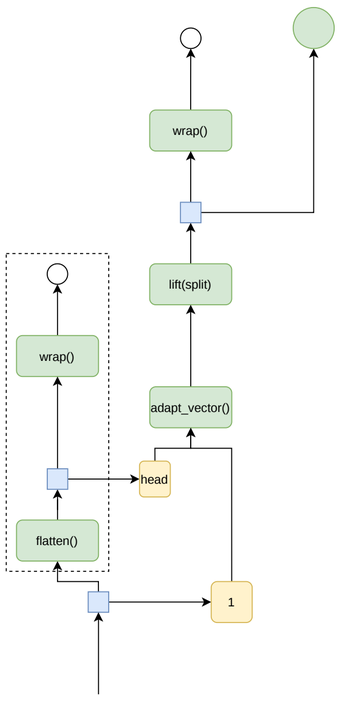

```
struct Wire
    branches::Vector{Wire}
end
```

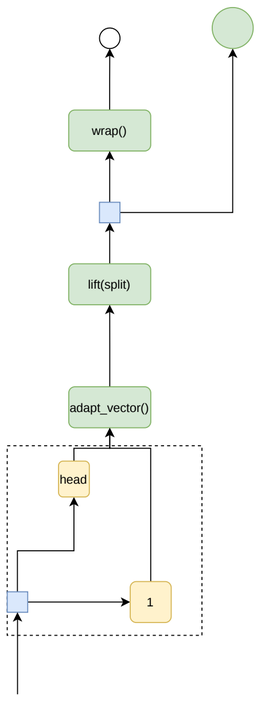


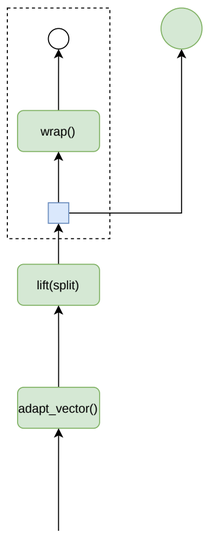


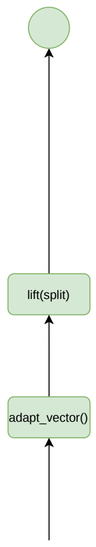


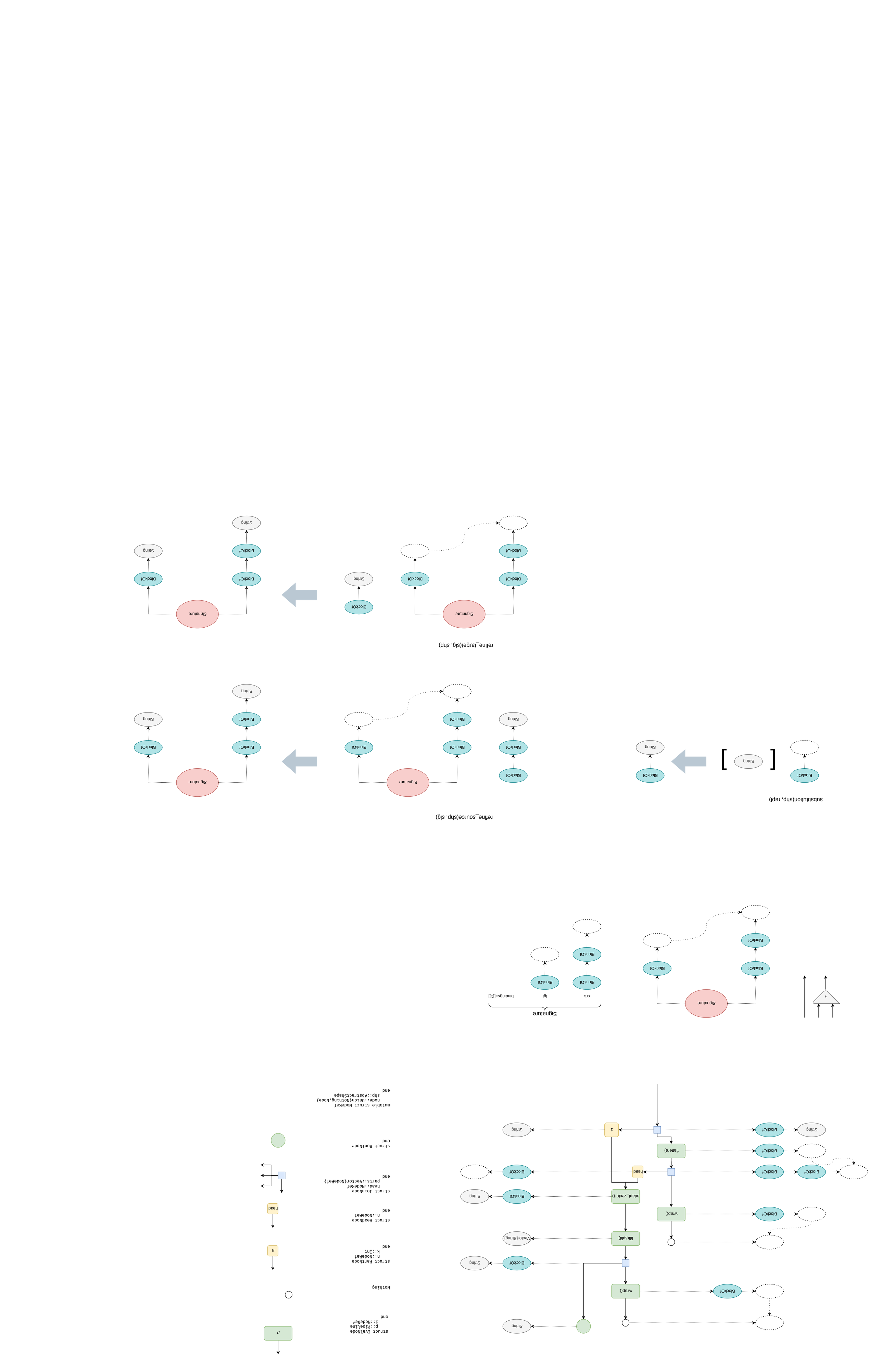



wrap()
lift(split)
wrap()
adapt_vector()
head
flatten()
1

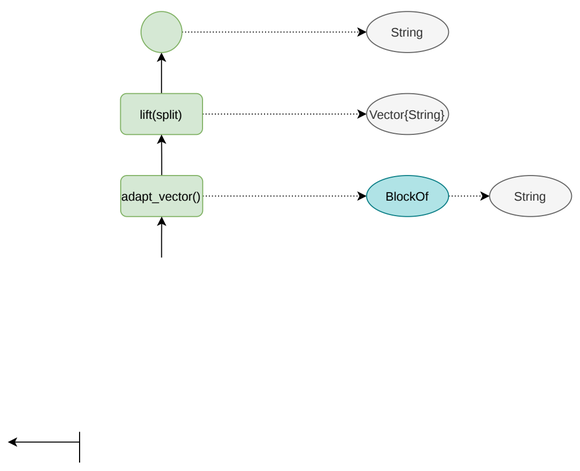

wrap()
lift(split)
adapt_vector()
head
1


wrap()
lift(split)
adapt_vector()

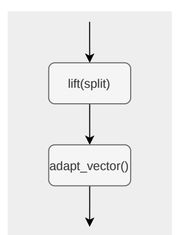

lift(split)
adapt_vector()


struct EvalNode
p::Pipeline
i::NodeRef
end
p
Nothing
struct PartNode
n::NodeRef
K::Int
end
n
struct HeadNode
n::NodeRef
end
head
struct JoinNode
head::NodeRef
parts::Vector{NodeRef}
end
struct RootNode
end
mutable struct NodeRef
node::Union{Nothing,Node}
shp::AbstractShape
end
refine_target(sig, shp)
Signature
src
tgt
bindings=[[1]]
BlockOf
*
refine_source(shp, sig)
substitution(shp, repl)
[
String
]
wrap()
flatten()
adapt_vector()
iff(split)
Vector{String}
1

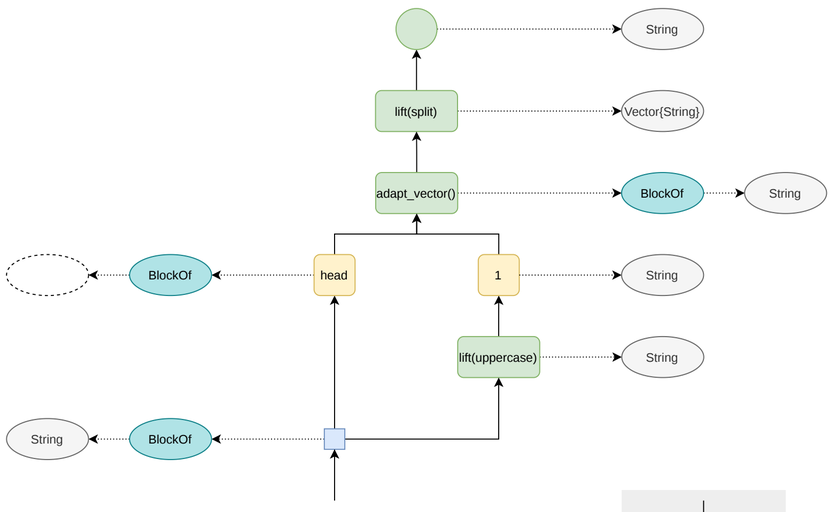
@query "Hello World" split(it) # after rewrite_unwrap

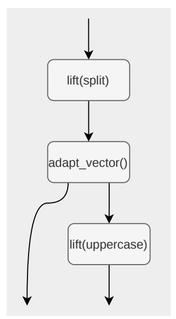


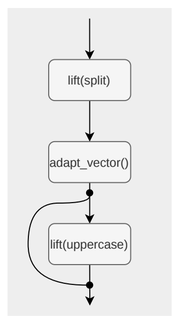
```
chain_of(
    lift(split),
    adapt_vector())
```



@query "Hello World" split(it).uppercase(it) # after rewrite_unwrap

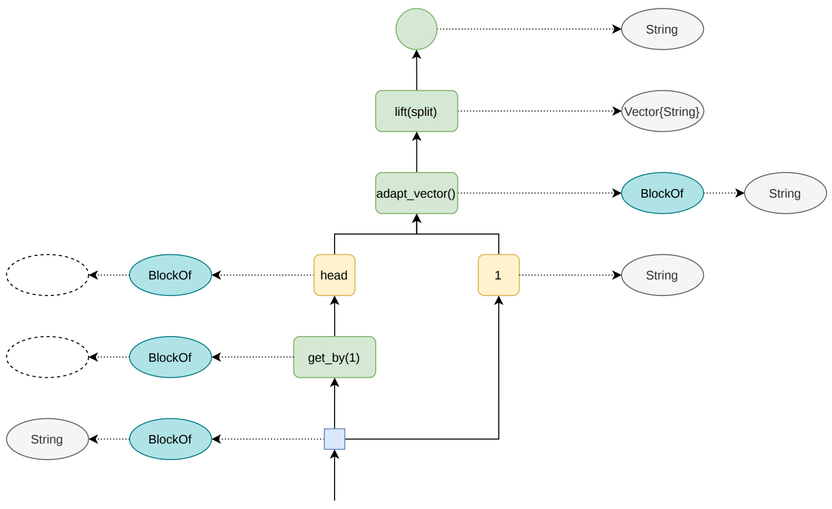


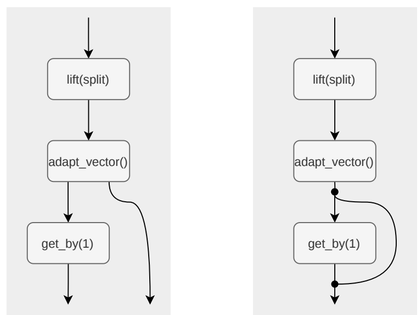





```
chain_of(
    lift(split),
    adapt_vector(),
    with_elements(lift(uppercase)))
```

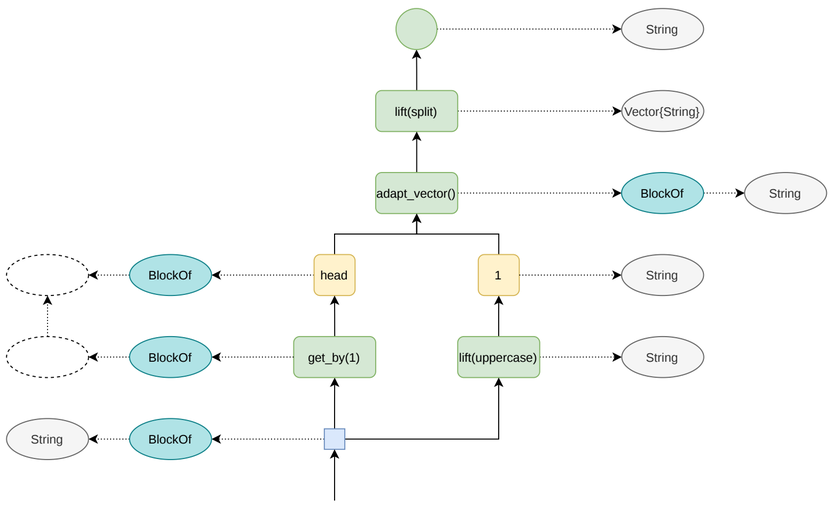
@query "Hello World" split(it).first() # after rewrite_unwrap

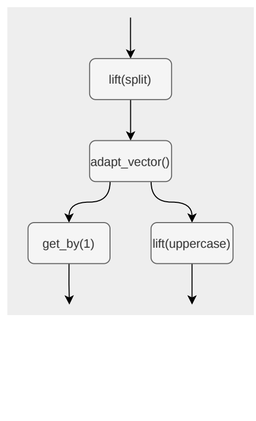


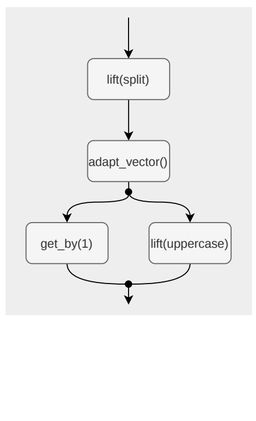



```
chain_of(
    lift(split),
    adapt_vector(),
    get_by(1, x0to1))
```

@query "Hello World" split(it).uppercase(it).first() # after rewrite_unwrap

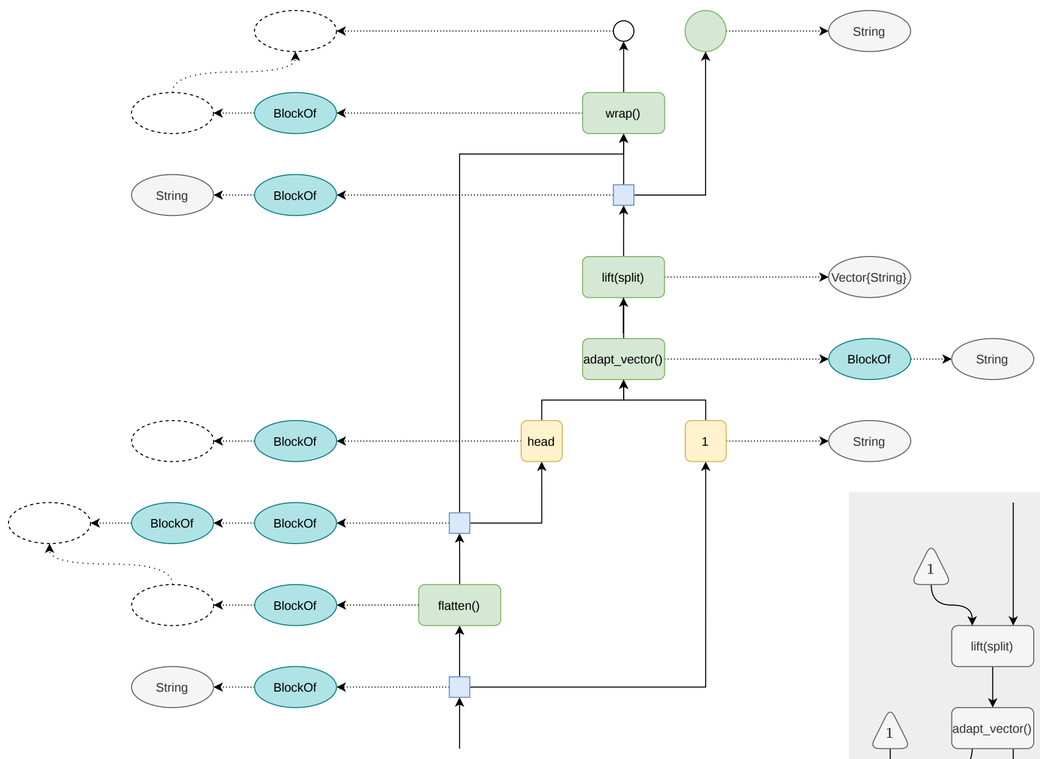


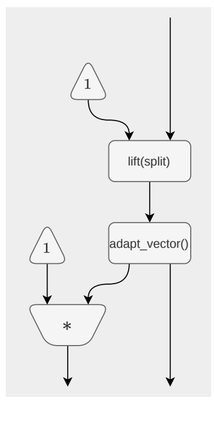


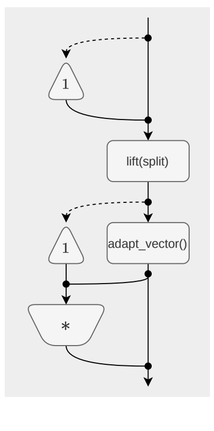



```
chain_of(
    lift(split),
    adapt_vector(),
    get_by(1, x0to1),
    with_elements(lift(uppercase)))
```

@query "Hello World" split(it)

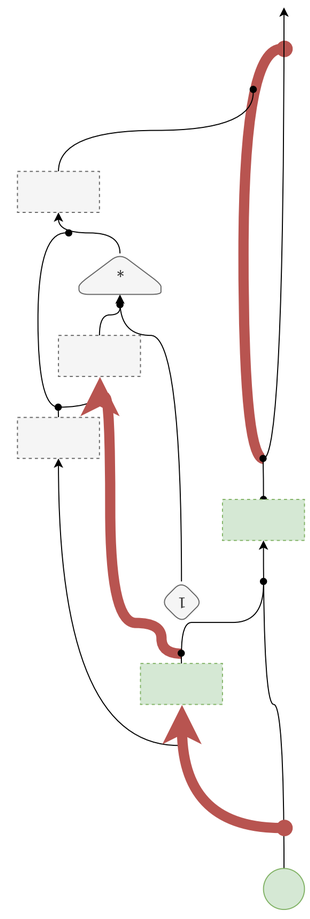






q
p
1
head

q
p
p


*
1