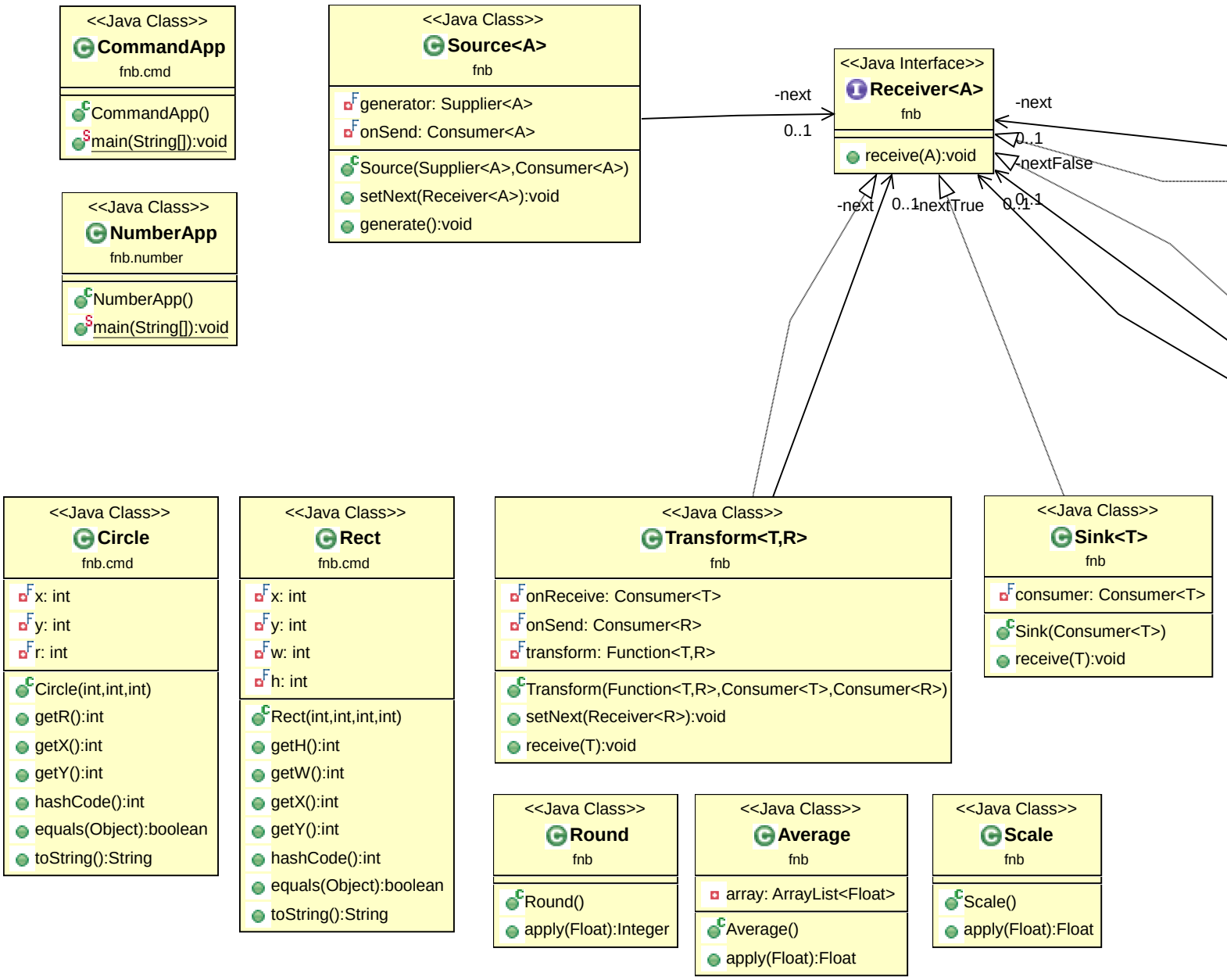
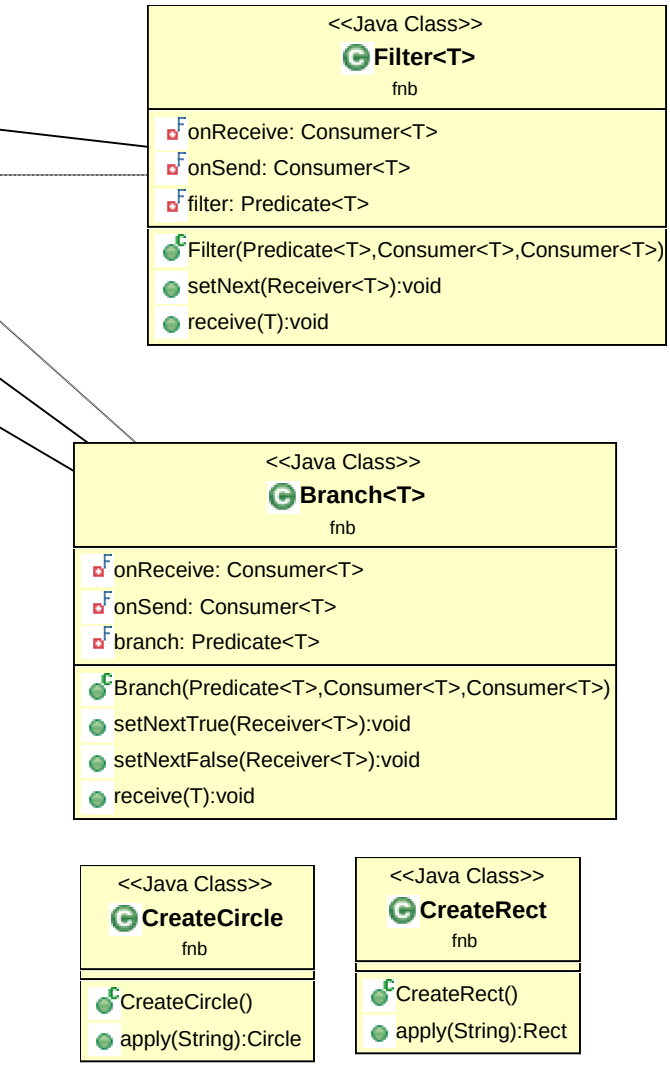

<<Java Class>>
CommandApp
fnb.cmd
CommandApp()
main(String[]):void
<<Java Class>>
NumberApp
fnb.number
NumberApp()
main(String[]):void
<<Java Class>>
Source<A>
fnb
generator: Supplier<A>
onSend: Consumer<A>
Source(Supplier<A>,Consumer<A>)
setNext(Receiver<A>):void
generate():void
-next
0..1
<<Java Interface>>
Receiver<A>
fnb
receive(A):void
-next
0..1
nextFalse
-next
0..1
nextTrue
0..1
<<Java Class>>
Circle
fnb.cmd
x: int
y: int
r: int
Circle(int,int,int)
getR():int
getX():int
getY():int
hashCode():int
equals(Object):boolean
toString():String
<<Java Class>>
Rect
fnb.cmd
x: int
y: int
w: int
h: int
Rect(int,int,int,int)
getH():int
getW():int
getX():int
getY():int
hashCode():int
equals(Object):boolean
toString():String
<<Java Class>>
Transform<T,R>
fnb
onReceive: Consumer<T>
onSend: Consumer<R>
transform: Function<T,R>
Transform(Function<T,R>,Consumer<T>,Consumer<R>)
setNext(Receiver<R>):void
receive(T):void
<<Java Class>>
Sink<T>
fnb
consumer: Consumer<T>
Sink(Consumer<T>)
receive(T):void
<<Java Class>>
Round
fnb
Round()
apply(Float):Integer
<<Java Class>>
Average
fnb
array: ArrayList<Float>
Average()
apply(Float):Float
<<Java Class>>
Scale
fnb
Scale()
apply(Float):Float


<<Java Class>>
Filter<T>
fnb
onReceive: Consumer<T>
onSend: Consumer<T>
filter: Predicate<T>
Filter(Predicate<T>,Consumer<T>,Consumer<T>)
setNext(Receiver<T>):void
receive(T):void
<<Java Class>>
Branch<T>
fnb
onReceive: Consumer<T>
onSend: Consumer<T>
branch: Predicate<T>
Branch(Predicate<T>,Consumer<T>,Consumer<T>)
setNextTrue(Receiver<T>):void
setNextFalse(Receiver<T>):void
receive(T):void
<<Java Class>>
CreateCircle
fnb
CreateCircle()
apply(String):Circle
<<Java Class>>
CreateRect
fnb
CreateRect()
apply(String):Rect