

Stone Paper Scissors

The case of results

S.Ducasse, L. Fabresse, G. Polito, and P. Tesone



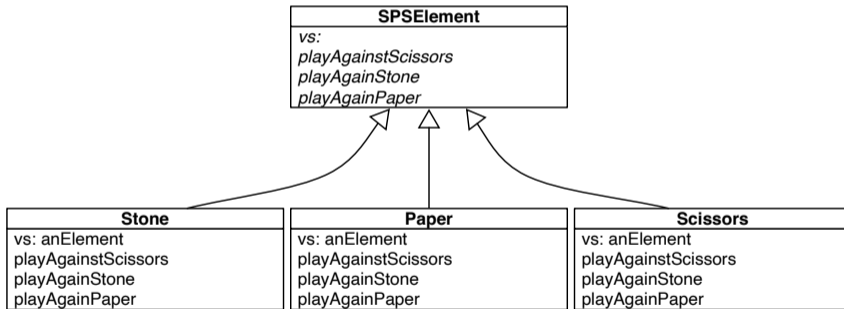
Goals

- Think about results
- What to do with them



Remember

> Stone new vs: Paper new
#paper



What should we return?

- What symbols, numbers?
- Consequences?



Returning numbers

```
StonePaperScissorsTest >> testPaperIsWinning  
self assert: (Stone new vs: Paper new) equals: -1
```

```
StonePaperScissorsTest >> testStoneAgainstStone  
self assert: (Stone new vs: Stone new) equals: 0
```

```
StonePaperScissorsTest >> testStoneIsWinning  
self assert: (Stone new vs: Scissors new) equals: 1
```



Returning number analysis

- We have to know the message sent to know how to interpret the number

```
> Stone new vs: Paper new
```

```
-1
```

```
> Paper new vs: Stone new
```

```
1
```

- Here paper is winning but it got different results!



Returning symbols analysis

```
> Stone new vs: Paper new  
#paper
```

```
> Paper new vs: Stone new  
#paper
```

- With symbols we know who won
- We will have to compare to do something

```
(aPlayer vs: anotherPlay) = #draw  
ifTrue: [ ... ]
```



Alternate solution

```
Paper new vs: Paper new  
onDraw: [ Game incrementDraw ]  
onReceiverWin: [ ]  
onReceiverLose: [ ]
```

```
Paper >> playAgainstStone: aStone  
onDraw: aDrawerBlock  
onReceiverWin: aWinnerBlock  
onReceiverLose: aLoserBlock
```

^ aWinnerBlock value

It feels that there are too many parameters.



Simply with objects

Paper new vs: Paper new withResultHandler: ResultHandler new

Paper >> playAgainstStone: aStone withResultHandler: aResultHandler
^ aResultHandler paperWon

- It feels like the "Do not ask tell"
- We can have different result handlers
- Feel free to experiment to see how it goes



Stepping back

- Symbols look better than numbers for returning information for this case but there are numerous cases where a number is definitively better
- Creating result handlers can be heavy
- Context and scenarios often give you information to see
 - where to invest
 - what is worth to support reuse



Conclusion

A design solution is often dependent on a context

- Try alternate solutions, compare them
- Exercise your design taste



Produced as part of the course on <http://www.fun-mooc.fr>

Advanced Object-Oriented Design and Development with Pharo

A course by

S.Ducasse, L. Fabresse, G. Polito, and P. Tesone



Except where otherwise noted, this work is licensed under CC BY-NC-ND 3.0 France
<https://creativecommons.org/licenses/by-nc-nd/3.0/fr/>