SIMPLICITY IN DESIGN

Rob Norman

"Design adds value faster than it adds costs."
Joel Spolsky

Takeaways so far

- BDD / TDD
- Transformation Priority Premise
- Object Calisthenics

BDD / TDD

- Reg, Gren, Refactor
- Determine the degrees of freedom then test one at a time
- Triangulation, do the very simplest and straight forward thing to satisfy the current tests, until we have enough tests in place to drive out a more generic implementation
- Rule of Three, try to see pattens in duplicated code when we have at least three repetitions

Transformation Priority Premise

- Useful to think about or refer back to whilst looking for ways to refactor.

# TRANSFORMATION	STARTING CODE	FINAL CODE
1 {} => nil		return nil
2 nil => constant	return nil	return “1”
3 constant => constant+	return “1”	return “1” + “2”
4 constant => scalar	return “1” + “2”	return argument
5 statement => statements	return argument	return arguments
6 unconditional => conditional	return arguments	if(condition)return arguments
7 scalar => array	dog [dog, cat]	
8 array => container	[dog, cat]	{dog = “DOG”, cat = “CAT”}
9 statement => recursion	a + b	a + recursion
10 conditional => loop	if(condition)	while(condition)
11 recursion => tail recursion	a + recursion	recursion
12 expression => function	today - birthday	CalculateAge()
13 variable => mutation	day	var day = 10; day = 11;
14 switch case		

Object Calisthenics

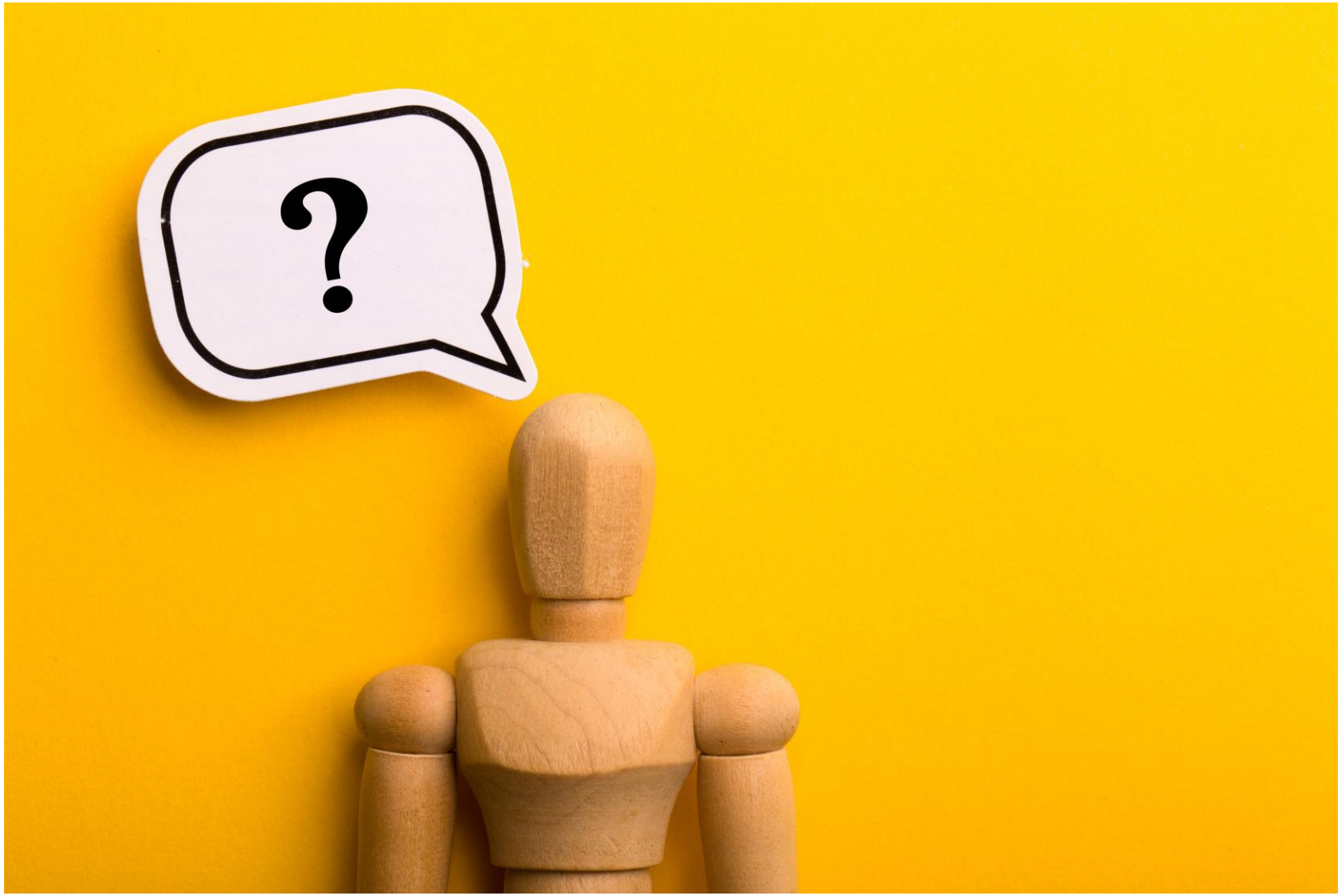
- Only one level of indentation per method
- Don’t use the ELSE keyword
- Wrap all primitives and strings
- First class collections (wrap all collections)
- Only one dot per line
- No abbreviations
- Keep all entities small [10 files per package, 50 lines per class, 5 lines per method, 2 arguments per method]
- No classes with more than two instance variables
- No public getters/setters/properties

Object Calisthenics

Heuristics

- Tell don’t ask
  - Tell the object what you want and let it figure out how to do it
  - As the caller you should not be making decisions based on the state of the called object that result in you then changing the state of that object, that smells of a leaky abstraction

- Law of Demeter
  - Each unit should have only limited knowledge about other units: only units "closely" related to the current unit.
  - Each unit should only talk to its immediate friends; don't talk to strangers.



Questions and Discussions