read & think

- A1 due next Tues — extra office hours TBA
- E3 is up; grading runs today — you can write unit tests
- SLOG entry #1 after A1

CSC148 fall 2013

more recursion, testing

week 4

Danny Heap
heap@cs.toronto.edu
BA4270 (behind elevators)
http://www.cdf.toronto.edu/~heap/148/F13/
416-978-5899

September 30, 2013

Outline

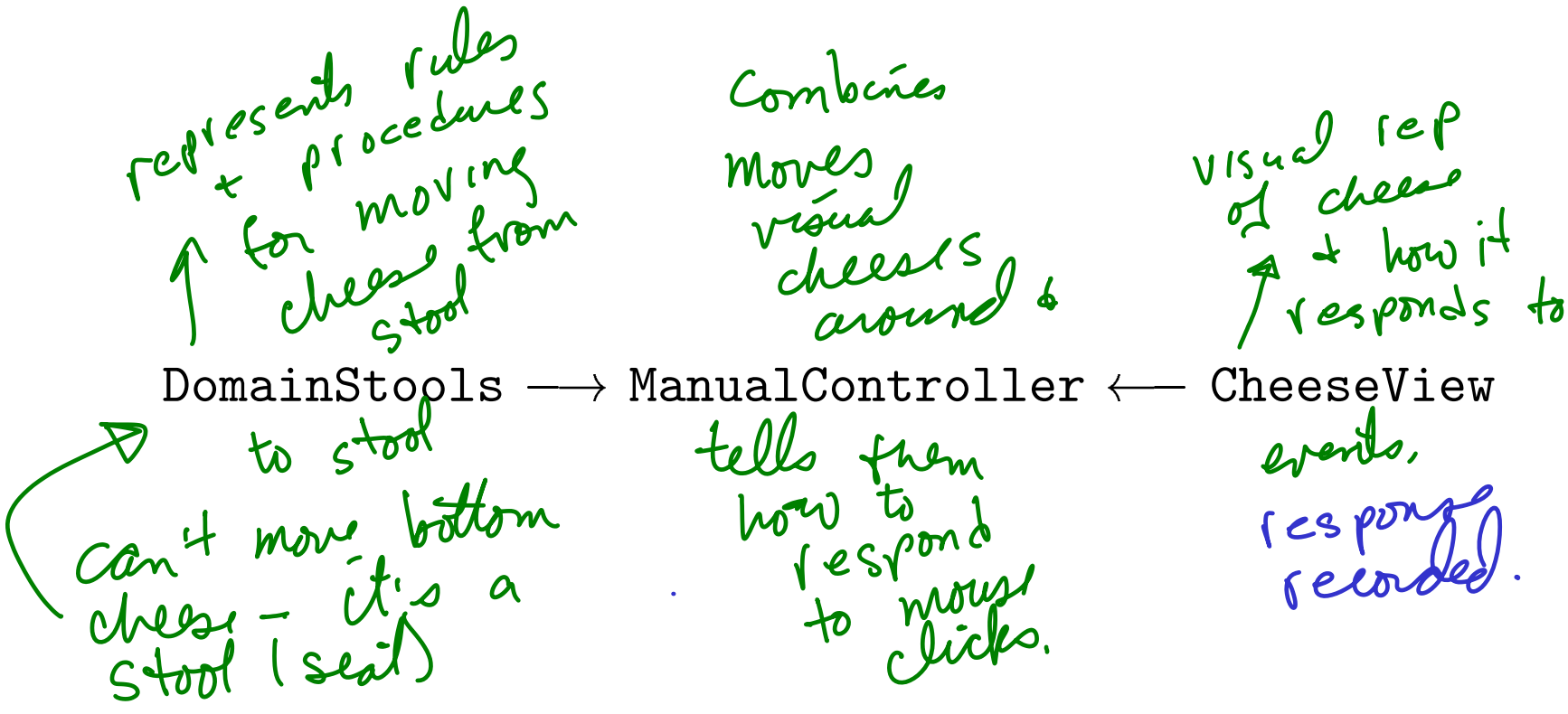
Class design for cheese

Recursion on nested lists

Testing, big and small

Separation of concerns

Tk inter docs version 8.5



nesting depth of list

[1, [2,3], 4, 5] → n_depth 2

[1,2,3] → n_depth 1

"hi" → 0

Define the `nesting-depth` of `L` as 1 plus the maximum nesting depth of `L`'s elements if `L` is a list, otherwise 0.

- the definition is almost exactly the Python code you write!

  that's beauty of recursion!

- start by writing `return` and pythonese for the definition:

```
return 1 + max([nesting_depth(x) for x in L]) if
```

  → isinstance(L, list) else 0

- deal with the special case of a non-list

maximum number in nested list

Use the built-in `max` much like `sum`

- how would you find the max of non-nested list?

  `max(...)`

- how would you build that list using a comprehension?

  `max([...])`

- what would you do with list items that were themselves lists?

  `max([rec_max(x) ...])`

- get some intuition by tracing through flat lists, lists nested one deep, then two deep...

before and after coding:

Test your docstring examples automatically:

```
if __name__ == '__main__':
    import doctest
    doctest.testmod()
```

For more thorough testing, use `unittest`