# Project Notes:

## Meeting Notes 07/29/15

* Programming: Focus on two sections in the duration of the course: early and middle
  + Early section: Very simple movement commands; autograde with python; give skeleton and procedure instructions; focus on reporters, function passing
    - Start with deterministic procedures and move to stochastic (seed runs)
      * Mushroom model, flocking?, Butterfly
  + Mid section: More complex model building; videos with instructions; Forest Fire, Sheep v. Wolf Predation
    - Make mistakes
    - Look at function passing
    - Look at random
    - Do some model analysis – Reporters, plots, behaviorspace, data plotting?
* Two folders – Papers (“wisdom literature” – see Phil 25) and models (~20 important/iconic models)
  + Philosophy – Phil 25 papers
  + Models – Schelling, Sugarscape, flocking, predation/Lottka-Volterra (comparison),
    - physics?
* Think about any speakers?
* Email directory of code to Prof. Kimbrough
* How to get Khan academy-style window with side-by-side video and coding

## Progress

* hw1 file and writeup in progress
  + began by hard-coding the necessary data collection in python.
  + Note: possible use of export-world (a pain to figure out the csv stuff, and too big)
  + switched to creating a .txt answer file (in csv format) with the necessary field values at each hw set, problem, and stage
    - Problem: This may be too inflexible, even with seeding (eg. what if someone uses (random-float 32) – 16 instead of random-xcor to set random start position? Does this impact the location? – weirder examples that would mess up the seed)
  + Finished coding the framework for the static answer file autograder, including fields for the grading/recording procedures
* Next:
  + error out more gracefully when no answer files
  + more sophisticated grading for errors, i.e. point system, also encoded in a dictionary – can add in a grading method that calls check\_hw\_answers; will also need more sophisticated records than just whether passed each test wholly
  + need approximate equality checking – may want to make this method more complex, deal with scaling of some kind
  + potential need for more dynamic grader w/ regard to random seeding – see above problem example
  + possible gui? – would only need two buttons + folder navigation; might be nice.