Slide 1

Administrivia

- Reminder: Homework 3 code due next Tuesday.

Slide 2

Homework 3 Hints — General

- Remember that most game framework interfaces and classes are generic, so to use them you should supply two "type parameters" (your block and entity interfaces). Why is it written that way? should start to become clearer with this assignment.
- Two groups of methods to define:
  - Methods of two framework interfaces, `Player` and `GameEntity`. Called once per game tick (normally — but you can have it called less often).
  - Methods of appropriate listener interface(s). Called when human player provides input.
- Think about what variables you need — in general, if there's something that's part of the object's "state" and needs to be used by more than one method, it should be an instance variable.

Slide 3

Homework 3 Hints — Drawing Things

- Individual blocks and entities: What's drawn is controlled by `getImage`. Blocks all scaled to same size. Entities scaled based on `partialSizeX/Y`. Positions of entities based on locations.
- "Partial"? The framework allows you to define, for the purposes of moving and scaling, a "partials in whole" number (to allow moving in fractions-of-a-block units).

Slide 4

Homework 3 Hints — Drawing Things, Continued

- Laying out screens: You can do this in code (probably in your screen class) or using the "screen editor" (brief description and links to more info in writeups for Homeworks 2 and 3).

  Potential "gotcha": If you set "partials in whole" to a non-default value (in your game setup class), and you want to use the screen editor, you also need to set "partials in whole" in your screen class.
- Notice / recall that not everything has to be part of the playing field: Your game can also include "panels" on any or all four sides. (We won't add those until Homework 6; for now you could consider just printing information to the console.)

Slide 5

Homework 3 Hints — Responding to Input, Moving Around

- Game ticks and keyboard/mouse events aren't particular in synch.
- So listener methods should probably just record information, to be processed by `update` method.
- Look at documentation of (Java library) listener interfaces to know what methods to write. Follow links to find out about other useful classes (e.g., `KeyEvent`).
- "Move" by changing location. Useful methods in (framework) `Location` class.

Slide 6

Homework 3 Hints — Interacting With Blocks

- At least some code for interacting with blocks goes in player classes. Similarly for other entities. However, good use of block hierarchy can help.
- Example — how do you not go through walls?

  (Contrast the "old way" using `instanceof` versus the "new way" using polymorphism and your interfaces.)
- Interacting with other entities starts in Homework 4, and is done in a similar (but not exactly the same) way.

Slide 7

Abstract Data Types

- "Abstract data type" (ADT) is defined as
  - A set of values.
  - A set of operations on those values.
- In other words — something that stores data (in an unknown form) and provides a standard interface for dealing with it.

Slide 8

Stack ADT

- Value — list of elements.
- Operations — push, pop, "empty?"
- Implementing this? might be a good example of
  - Defining a (generic) interface.
  - Writing a class to implement it (using arrays — for, um, fun? practice?).
  - Working with exceptions.

Slide 9

Queue ADT

- Value — list of elements.
- Operations — enqueue, dequeue, "empty?"
- We could implement similarly to what we did for stacks . . .

Slide 10

Minute Essay

- How did the midterm compare to what you expected? with regard to length, difficulty, topics . . .