Class Relationships

Class Relationships

- In systems with multiple classes, it can become difficult to keep track of relationships.
- eg. the `Student` class requires on the `Course` class to work.
- There are many ways classes can interact.
  - Details of the interactions are part of the design.

Dependancy

- It's very common for one class to use another.
  - methods use a class to temporarily store/manipulate information
  - an instance variable stores data with another class
  - arguments are passed of another class type
  - etc.
- … any case where a class needs another to compile or run.

Aggregation

- Special type of dependency.
- A dependant class is an aggregate if it is "part of" the larger class.
  - … not just used in the implementation, but really part of the actual "object".
  - ie. the real object that we're modeling with the class has one of these as a part of it.
- eg. part of a "hand" is a "card".

Aggregation

- Exactly what is aggregation?
- Generally: if a class is used as an instance variable, it's an aggregate.

```
class Player {
    Hand h;            <-- instance variable: aggregation
    public takeTurn() {
        String input;  <-- method variable: dependence,
    }                      but not aggregation
}
```

UML Diagrams

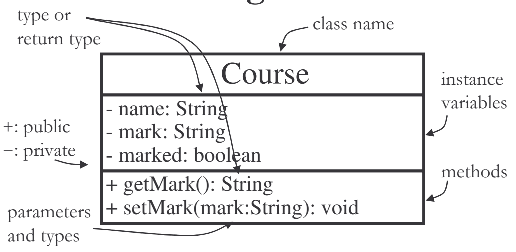
UML

- Unified Modeling Language
  - general methods to model/design/document software and other structures
  - commonly used to design object-oriented systems
- There are several different types of UML diagrams.
  - … including "class diagrams"

Class Diagrams

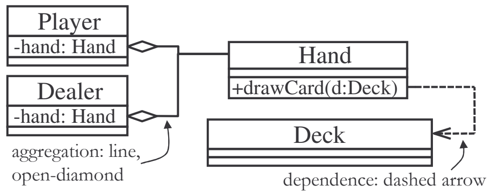
- Diagrams that are used to describe classes and class relationships
- Represents:
  - classes: instance variables, methods (and relevant types and arguments)
  - class relationships: dependence, aggregation, …

Drawing a Class



- Some details occasionally omitted (for clarity).
- Not Java syntax: independent of language

Class Relationships



- Other arrow types represent other dependencies.
- You won't be asked to draw these, but they might be used in future explanations.

Design & UML

- A full UML class diagram gives **a lot** of information about the design.
  - Creating one requires **very** careful planning.
- Probably too detailed for initial planning.
  - Could be done for low-level design or documenting.
  - Often, details will have to change during implementation, but don't plan it that way.
- Our blackjack design isn't really finished yet.