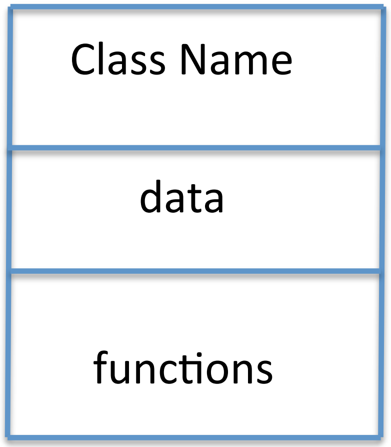
Finding Classes

What is a Class?

- Should be self-contained
  - Should have necessary data
  - Should have ALL functions that access that data
- Typically drawn graphically as:



Class Identification

- Decompose domain into classes
  - Self-contained, has all data and functions for task
  - Focused on specific purpose
- Typically you can brainstorm possible classes
  - Take the list and look for has-a or is-a relations
  - Assign functions to entities or item
- Develop the class hierarchy
- Much of this may be done if expanding an existing system

Purpose of the Class

- Data and functions determined by class should do or represent
- What data is needed for this purpose?
- What operations are needed to achieve this purpose?

- Purpose can also be understood as the semantics of the class

What Data?

- Identify all information required for the class to achieve its purpose
- What could it be?
  - Location? Coordinates or room # or ???
  - State? In motion- vector, in-service, or ???
  - Container? Is it holding some thing or things ???
- All data should be private
  - Maintains encapsulation

OSU Oregon State University

What Functions?

- To achieve the purpose of the class what actions are required?
  - Get and Set are not always required
- Which actions are called from outside the class? They should be public.
- Which actions are internal to the class? They should be private or protected.
- Identify auxiliary, internal actions required

Put It All Together

- Now you can outline each of your classes
- Determine the descriptive class name
- Confirm the data and actions required
- Review internal and external actions
- Repeat for all proposed classes

Relations

- Can be the "tricky" part of OOD
- Some are "obvious"
  - Waged_Employee is an Employee
  - Janitor is a Waged_Employee
- Some are less so
  - The class Job has-a person (i.e. a data element)
  - The class Person has-a job (i.e. a data element)
- Right or wrong depends on what you're doing

Inheritance

- So far you should just have been brainstorming
- Three things to look for now
  - Classes that overlap may need a common parent
  - Subclasses that don't overlap their parent(s)
  - After each step repeat to find new patterns
- When finished you will have the class diagram
- NOW it is time to start coding

Be Flexible

- As you identify classes do not be afraid to consider different ways to split up the domain or task
- As you analyze the situation you gain understanding and may realize there is another way
- First consider if it is better, if so change it
- It is a draft work with both until you know

Class Identification

- Identify potential classes for the domain
- Clearly state the purpose of the class
- Decide what data is needed for that purpose
- Decide what actions are needed
  - Decide if they are internal or external actions
- Develop the inheritance patterns/structure
  - Revise private to protected where appropriate