

Object-Oriented Design

Keith T. Weber, GISP
GIS Director, ISU



Topics

- During the balance of this semester, we will pursue and follow two learning threads
 - Object-relational databases
 - The Geo-Web
- These two threads are interwoven



To understand Object-Relational Databases...

- We need to understand both relational concepts and
- Object-oriented concepts (this week)

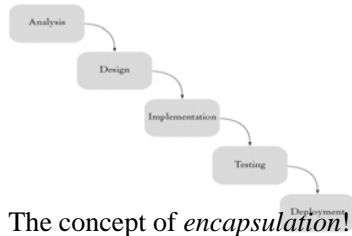
The Early Days...

- Computer programming from the caveman era



Why...Object-Oriented

- A brief history of computer programming...



The concept of *encapsulation!*

Today's Goals

- We will dissect "Object-Oriented" to learn what it really is
 - What is a class?
 - What is an object?
- Enable you to identify inheritance, aggregation, and dependency relationships between classes
- Understand class attributes and object properties
- Become familiar with new *terminology*

What is a CLASS?

- A *class* is a computer construct representing a concept bound in a cohesive package
 - Some are concrete (i.e. real world)
 - Bank account
 - Rental item
 - Database item
 - File
 - Others are abstract
 - Scanner
 - Stream
 - Math

Discovering CLASSES

- Simple Rule:
 - Look for *nouns* in descriptions
 - Obviously not all nouns are classes
 - But at least this method can allow one to create a list of *candidate classes*

What is an OBJECT

- An *instance* of a CLASS
- Contains meaningful data
- Concepts that occupy memory space at runtime are, according to the definition, objects
 - If not, they are CLASSES
 - For example: data type vs. double

A little Quiz...

- #1 Class or Object?



Dog

Dog is a generalization of Scooby-Doo

Scooby-Doo

A little Quiz (cont'd)...

- #2 Class or Object?

Animal

Dog

*Dog is a subclass of the Animal class
Animal is a generalization of Dog*

Scooby-Doo

The concept of *subclass!*

A little Quiz (cont'd)...

- #3 Class or Object?

Animal

Bird

Dog

The concept of *polymorphism!*

Questions...



Key Points

- Many classes already exist and are at our disposal when we design a database
 - We use *inheritance* to add capabilities to our projects
 - A *subclass* inherits from its *superclass*
 - i.e., a *child* inherits from its *parent*

Defining our CLASS

- After a class has been identified we can define:
 - The behavior of each class
 - Verbs = *methods*
 - The attributes of each class

Behavior

Relationships Between CLASSES

- We have learned about inheritance as a relationship between classes
 - There are 3 important relationships
 - Inheritance
 - Aggregation
 - Dependency

Inheritance

- *Is-a* relationship
- Relationship between a more general class (*superclass*) and a more specialized class (*subclass*)
- Every
 - savings account is a bank account
 - DVD rental is a rental

Aggregation

- *Has-a* relationship
 - Each Dog *has a* Paw (dog is not a generalization of paw!)
- Objects of one class contain references to objects of another class

Inheritance vs. Aggregation

- Often confused
- Questions?

Example

- Car is a Vehicle – Inheritance
- Car has a set of Tires – Aggregation

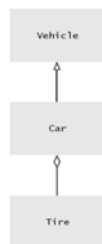


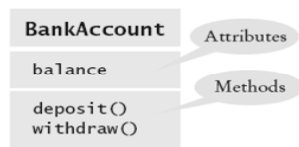
Figure 6
UML Notation for
Inheritance and Aggregation

Dependency

- Dependency occurs when a class uses another class' methods
- This is a *Uses* relationship
 - Example: an application may depend on the `scanner` class to read input

Attributes

Class Diagram



What type of Method behaviors are these?

Dog example

- Name of the class =
- Methods?
- Attributes?

Dog

Instantiate into object

- Three features characterize objects:
 - Identity: specific property settings have been made for the attributes of the class. This distinguishes it from other objects.
 - State: Describes the data stored in the object WHERE DID THIS COME FROM?
 - Behavior: describes the methods in the object's interface through which the object can be used.

Instantiating the Dog CLASS

- CLASS (DOG)
- Attributes (Properties)
 - NAME = Scooby-Doo
 - HEIGHT = 36
 - WEIGHT = 145
- Methods
 - [Uses] bark- "Rooby roo"
 - etc.



Scooby-Doo



Key Concepts

- Understand the difference between a CLASS and an OBJECT
- Understand new terms:
 - Encapsulation, polymorphism, superclass, subclass, behavior, attributes, instantiation
- Understand --and be able to differentiate-- the three types of behaviors

Questions...