File Edit View Help

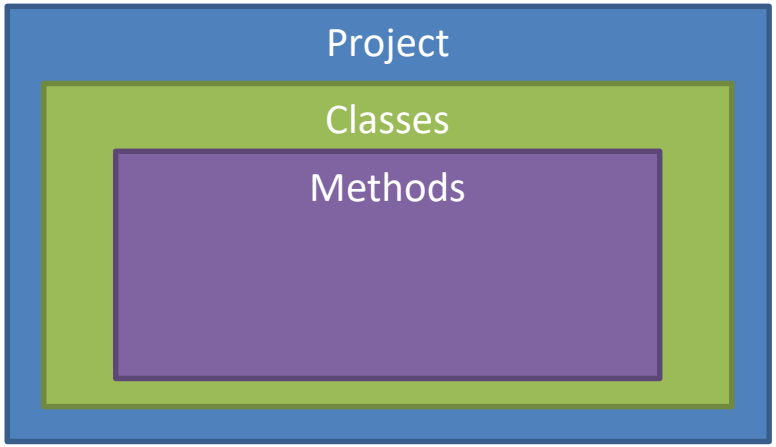
File Edit View Help

Classes and Objects
Part 02

Code Organization

- Organized and structured code helps to:
  - Reuse parts of code, so you use less statements
  - Quickly find bugs or errors
  - Easily add or extend functionality
- Java Organizes Software
  - First in Projects
  - Then in Classes
  - Then in Methods

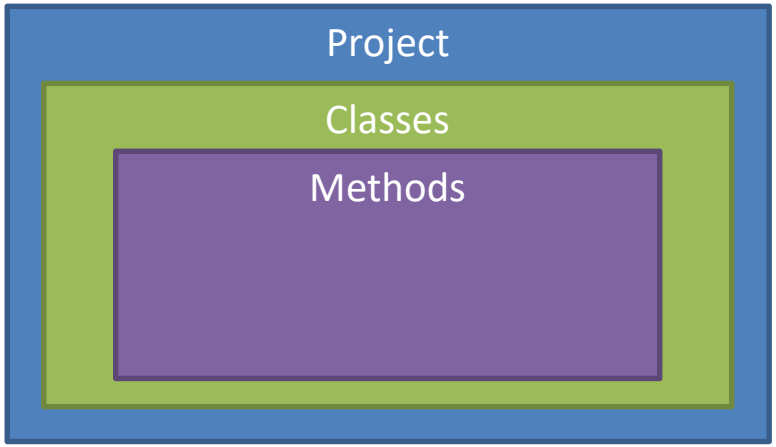
Java Software Structure



Classes

- Classes are a way that we can create *classifications* of “objects”
- Instances of a class are referred to as “objects”
- Classes provide a “blueprint” of a class of objects
  - Shared Qualities
  - Shared Characteristics
- Classes combine
  - Data (Attributes / Properties)
  - Methods (Actions)
- Think of Classes as *nouns*

Java Software Structure



Classes

Creating a Class in 7 Easy Steps!

1. Define the class
2. Create Properties
   1. Instance Variables
   2. Constants
3. Define Constructors
   1. Default
   2. Parameterized
4. Create Accessors for every Instance Variable
5. Create Mutators for every Instance Variable
6. Create other Methods
   1. equals()
   2. toString()
7. Use the Class to create Objects!

File Edit View Help

Example

Enumerations

- An enumeration (“enum”) is a special kind of Class that only contains constants
- Used when creating a type that only has a set number of potential values
- Good programming practice to create in a separate Java File (like classes)
- The constant values are separated using a comma (“,”) and values should be capitalized
- Declare an enum just like any other class
  - Does not require construction
- Access the defined values using the dot (“.”)

Defining an Enum

```
public enum <<identifier>>{
      <<Value00>>,
      <<Value01>>,
      …
}
```

Example

```
enum PetType {CAT, DOG, HAMSTER, HEDGEHOG,
ARMADILLO, TURKEY, OWL, ABOMINATION};
```

Enumerations

- An enumeration (“enum”) is a special kind of Class that only contains constants
- Used when creating a type that only has a set number of potential values
- Good programming practice to create in a separate Java File (like classes)
- The constant values are separated using a comma (“,”) and values should be capitalized
- Declare an enum just like any other class
  - Does not require construction
- Access the defined values using the dot (“.”)

Declaring and Using an Enum

```
//Delare Enum
<<enum identifier>> <<id>>;
//Using
<<id>> = <<enum identifier>>.<<Value>>;
```

Example

```
PetType type;
type = PetType.DOG;
```

Overloading Methods

- A method's identifier, return type, and parameters is called the "signature" or "definition"
- Overloaded Methods are methods with the same identifier's and return types, but different parameters
  - This is within the same class

Overloaded Method Example

```
public void giveComplement()
{
	System.out.println(this.name+" reacted with joy");
}
public void giveComplement(int c)
{
	for(int i=0;i<c;i++)
		this.giveComplement();
}
```