Programming with Objects

Jesse Liberty

@jesseliberty http://jesseliberty.com

Key Concepts

Inheritance

Polymorphism

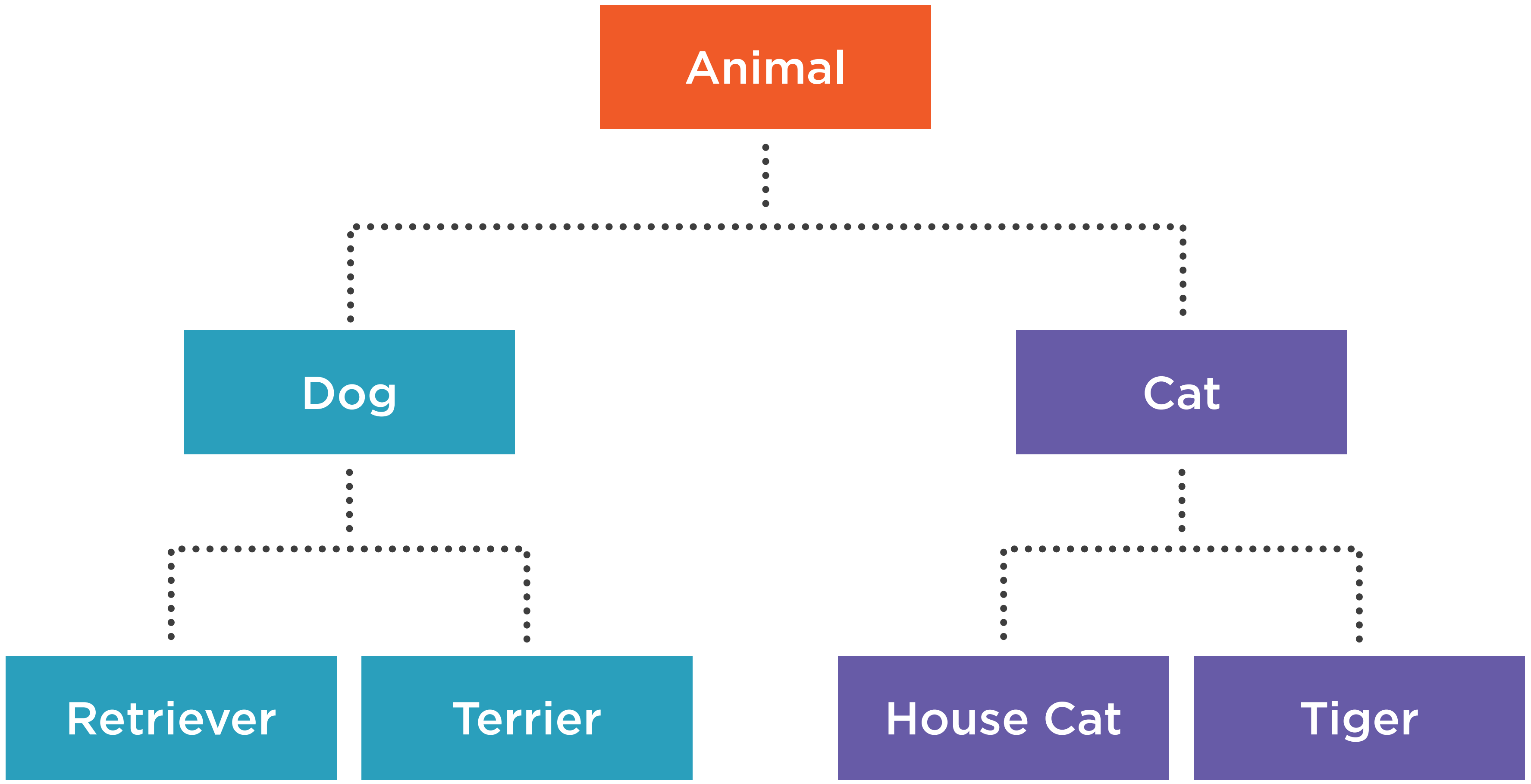
Encapsulation

Inheritance:

Classes may derive from existing classes

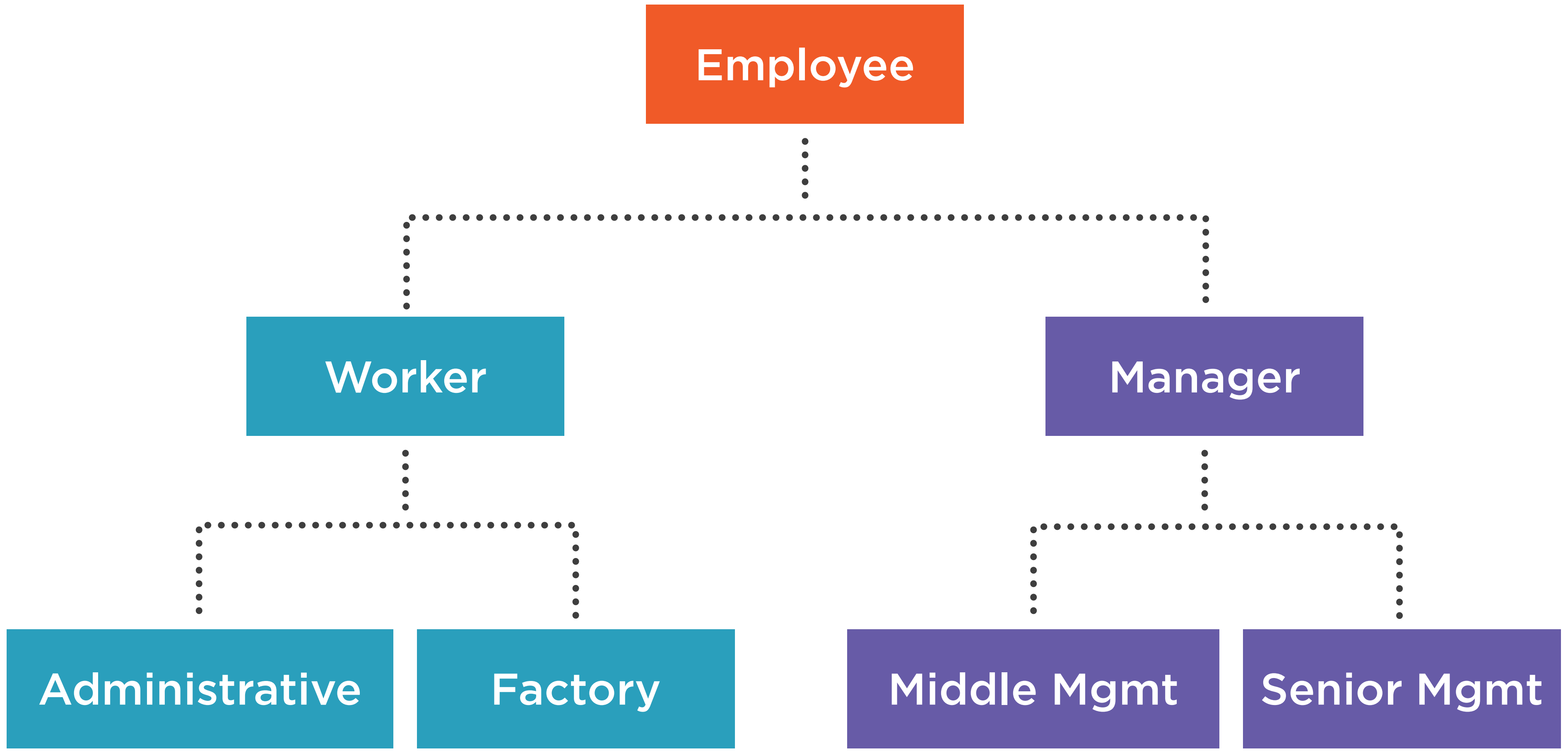
Derive

Specialize the “parent” class

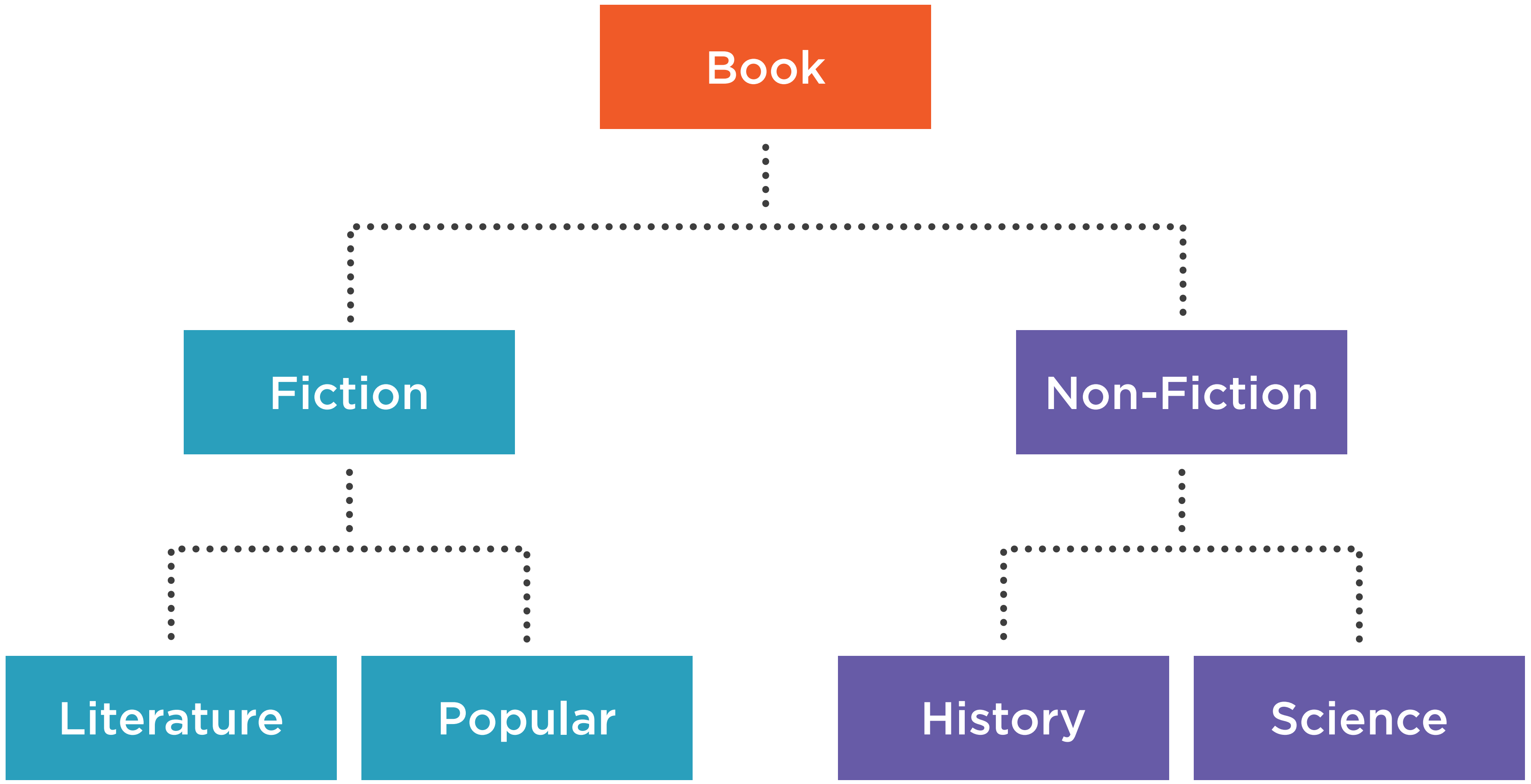

Animal
Dog
Cat
Retriever
Terrier
House Cat
Tiger

Parent Class

Generalization of the derived classes


Employee
Worker
Manager
Administrative
Factory
Middle Mgmt
Senior Mgmt

Inheritance creates an “ is – a ” relationship


Book
Fiction
Non-Fiction
Literature
Popular
History
Science

```
Class Book {
        // ...
}

Class Fiction : Book{
        // ...
}

Class NonFiction : Book{
        // ...
}
```

Derived Class Indicates Base Class with Colon

Containment

Classes can contain properties and fields

These can be of built in types and/or of user-defined types

Typically fields are used to support the private methods of the class

Typically properties are used to expose values to other classes

Containment creates a
“ has – a ” relationship

```
class Book {
        public TableOfContents tableOfContents { get; set; }
        public Index index{ get; set; }
        public virtual double Discount ();
}
```

Has-A Relationship

Demo

Inheritance

Polymorphism: Taking Many Forms

Method Overriding

Modifying a method in the derived class

```
Class Book {
          public virtual double Discount();
}

Class Fiction : Book{
          public override double Discount {

                    // other work here
          }
}
```

Virtual and Overridden Methods

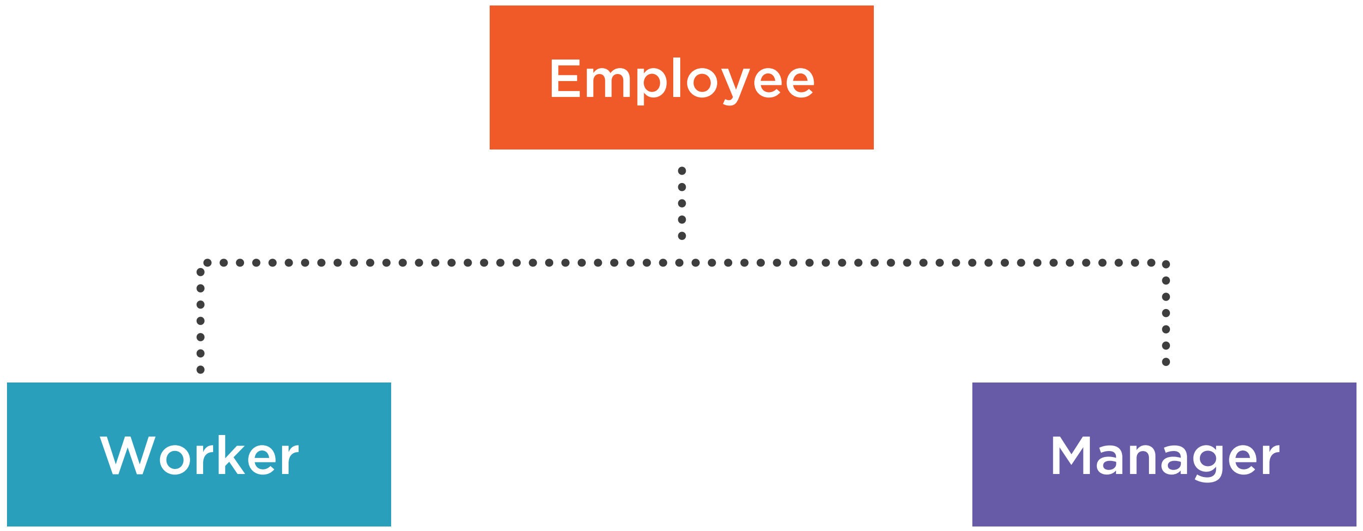
```
Class Book {
        public virtual double Discount();
}

Class Fiction : Book{
        public override double Discount() {
                        base.Discount();

                // other work here
        }
}
```

Chaining up to the Parent (base) Class

Derived and base classes can be treated polymorphically.


Employee
Worker
Manager

```
Employee joe = new Manager("Joe", true);

Employee bob = new Worker("Bob", 35.0);

Employee sally = new Worker("Sally", 27.50);
Employee joe = new Manager("Joe", true);
Employee bob = new Worker("Bob", 35.0);
Employee sally = new Worker("Sally", 27.50);

List<Employee> Employee = new List<Employee>();
Employees.Add(joe);
Employees.Add(bob);
Employees.Add(sally);
```

```
for (int i = 0; i < Employees.Count; i++) {
        Employees [i].TakeVacation();
        Console.WriteLine(Employees[i]);
}
```

Demo

Polymorphism

Encapsulation
each Class has a single responsibility

Encapsulation

Most of the internals of the class are private, with a few well-defined properties and methods that are public.

Demo

Putting it all together