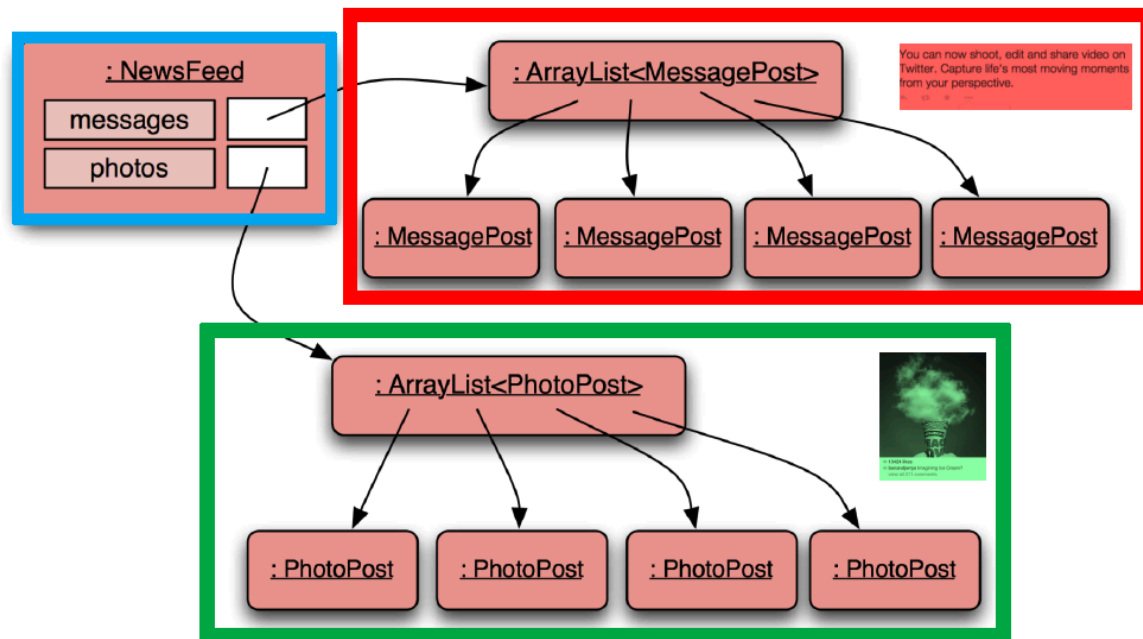




# No Inheritance

## Social Network V1 - Object model



# Review

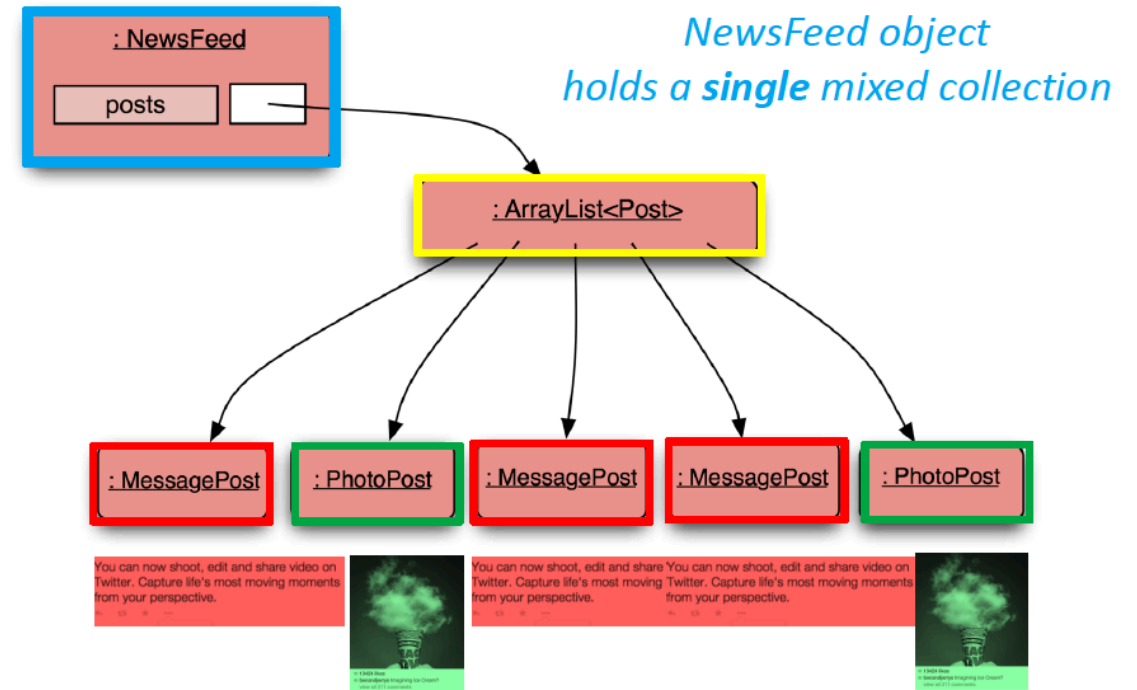
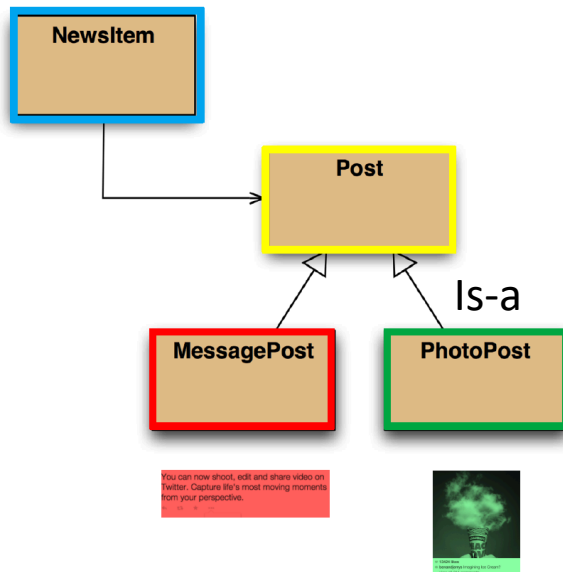
---

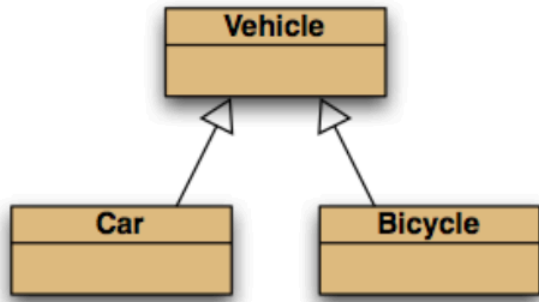
- Inheritance allows the definition of classes as extensions of other classes.
- Inheritance
  - avoids code duplication
  - allows code reuse
  - simplifies the code
  - simplifies maintenance and extending
- Variables can hold subtype objects.
- Subtypes can be used wherever supertype objects are expected (substitution).



# With Inheritance

## Social Network V2 - Class diagram





*subclass objects  
may be assigned to  
superclass variables*

```
Vehicle v1 = new Vehicle();  
Vehicle v2 = new Car();  
Vehicle v3 = new Bicycle();
```

- Object variables in Java are **polymorphic**

– They can hold objects of


- more than one **type**
- the declared **type**
- sub**types** (*of the declared type*).

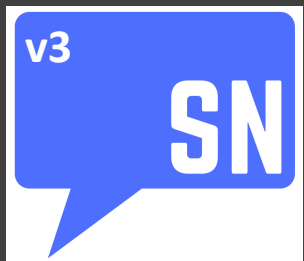
```
feed.addPost(photo);  
feed.addPost(message);
```

```
public class MessagePost extends Post
{
    private String message;

    /**
     * Constructor for objects of class MessagePost
     */
    public MessagePost (String author, String text)
    {
        super(author);
        message = text;
    }

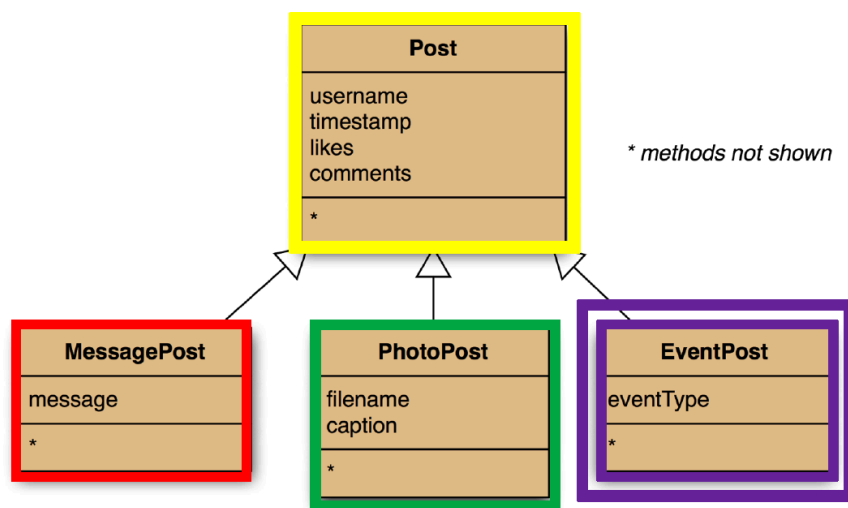
    // methods omitted
}
```



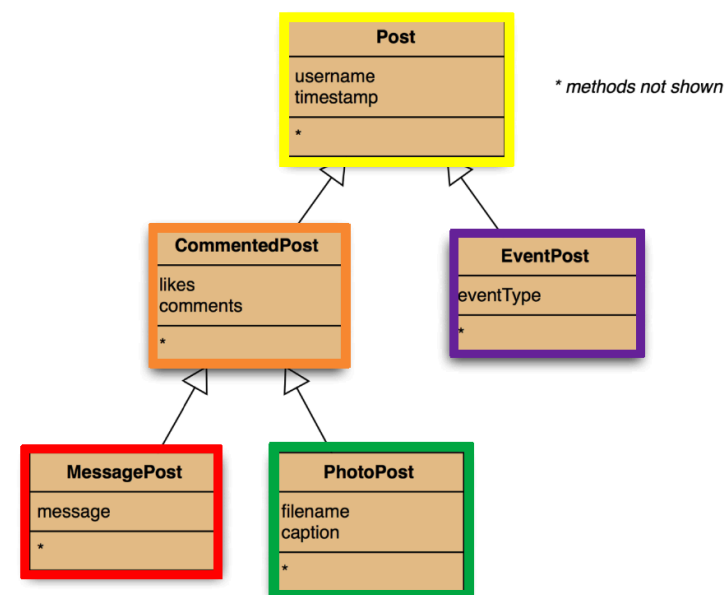


# With Inheritance

Social Network V3 - Adding more item types



Social Network V3 - Deeper hierarchies

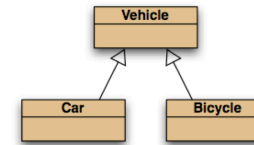


- a) Polymorphic Variables
- b) Polymorphic Collections
- casting,
  - wrapper classes,
  - autoboxing /unboxing

- i. more than one **type**
- ii. the declared **type**
- iii. **subtypes** (of the declared type).

### Casting

```
Vehicle v;  
Car c = new Car();
```



We can assign **subtype** to **supertype** (note arrow direction)!

```
v = c;
```

// correct (car is-a vehicle)

```
c = v;
```

But we cannot assign a **supertype** to **subtype** (cannot go against the arrows)!

// compile-time error!

Without (CASTING)

```
c = (Car) v;
```

//casting..correct (only if the vehicle really is a Car!)

### Wrapper classes

- Primitive types are not object types.  
Primitive-type values must be wrapped in objects, to be stored in a collection!★
- **Wrapper classes** exist for all primitive types:

primitive type

int  
float  
char  
...

wrapper class

Integer  
Float  
Character  
...

Note that there is no simple mapping rule from primitive name to wrapper name!



# Next

Explore Polymorphism further