Programming Fundamentals

Inheritance: Sub Classes

Produced By:
Mr. Dave Hearne, Dr. Siobhán Drohan, Mr. Colm Dunphy, Mr. Diarmuid O'Connor, Dr. Frank Walsh, Ms Mairead Meagher, Ms Siobhan Roche

SETU Ollscoil Teicneolaíochta an Oirdheiscirt
South East Technological University

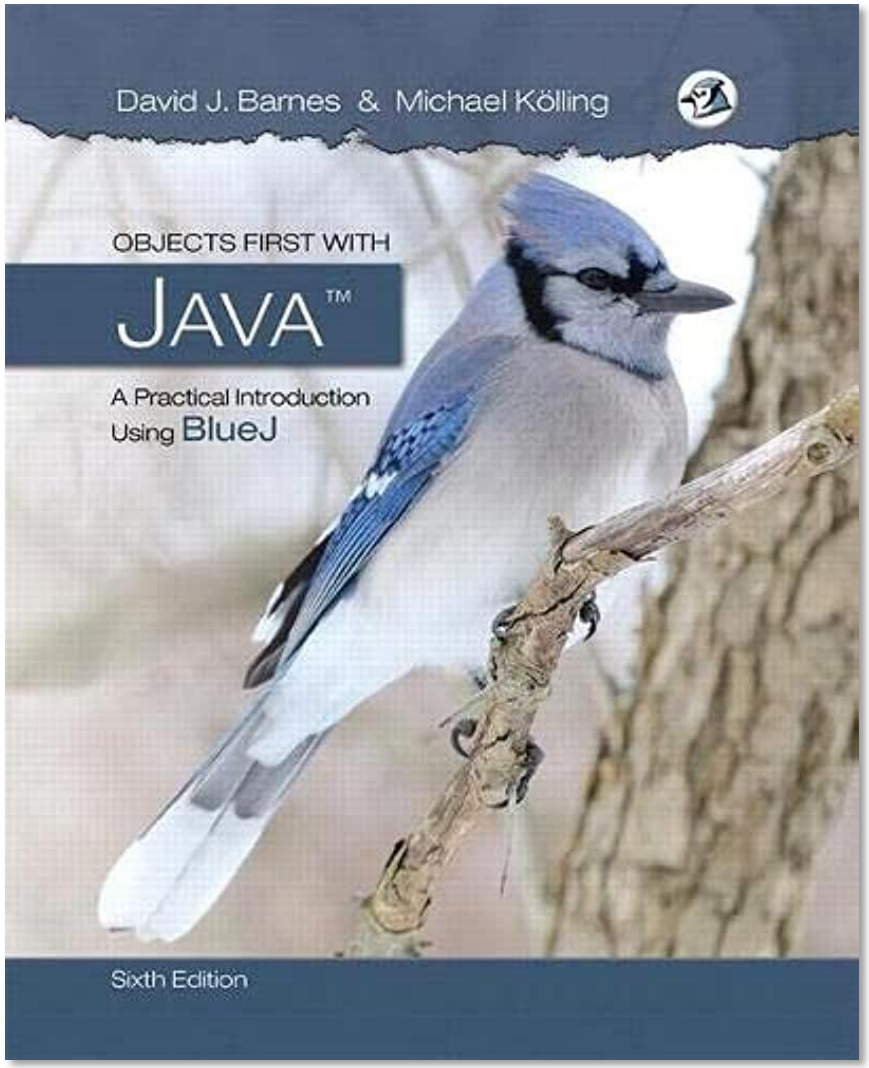
Agenda

- Inheritance Hierarchies
- Social Network v5
- Sub and Super Classes

Lectures & Labs

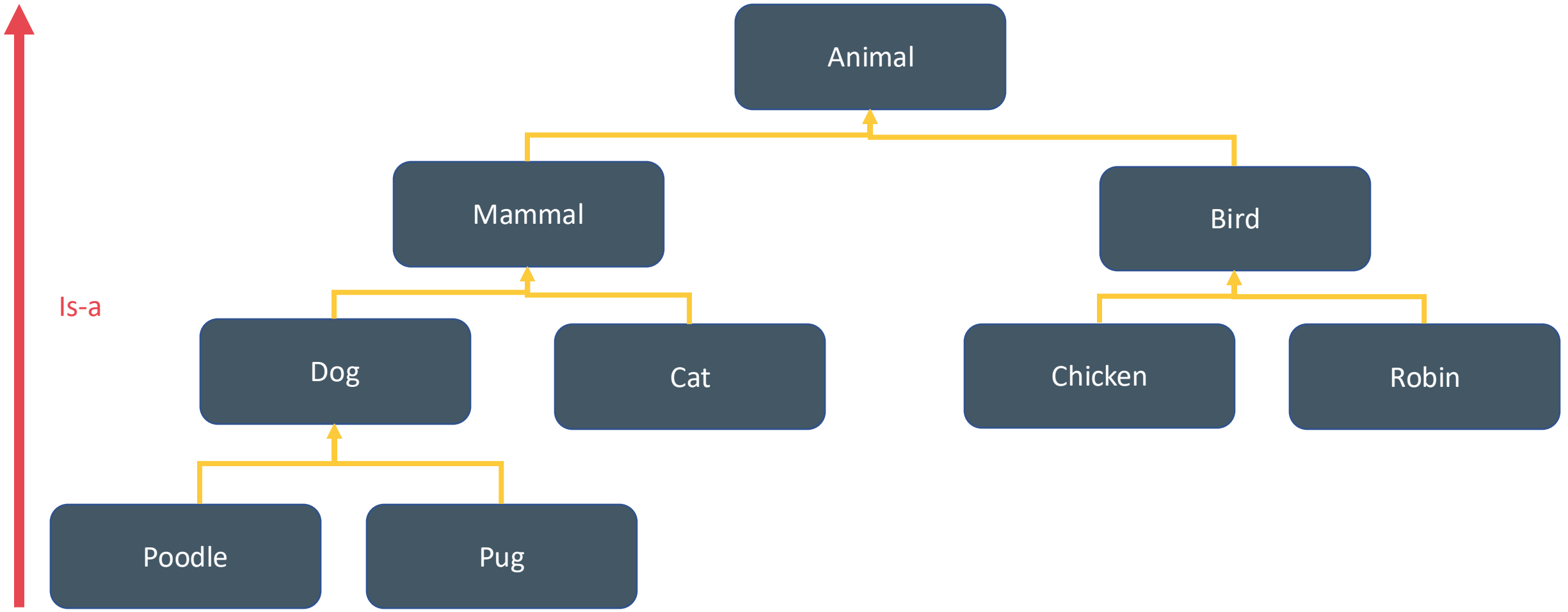
- This weeks lectures and labs are based on examples in:
- **Objects First with Java**

  - A Practical Introduction using BlueJ, © *David J. Barnes, Michael Kölling*



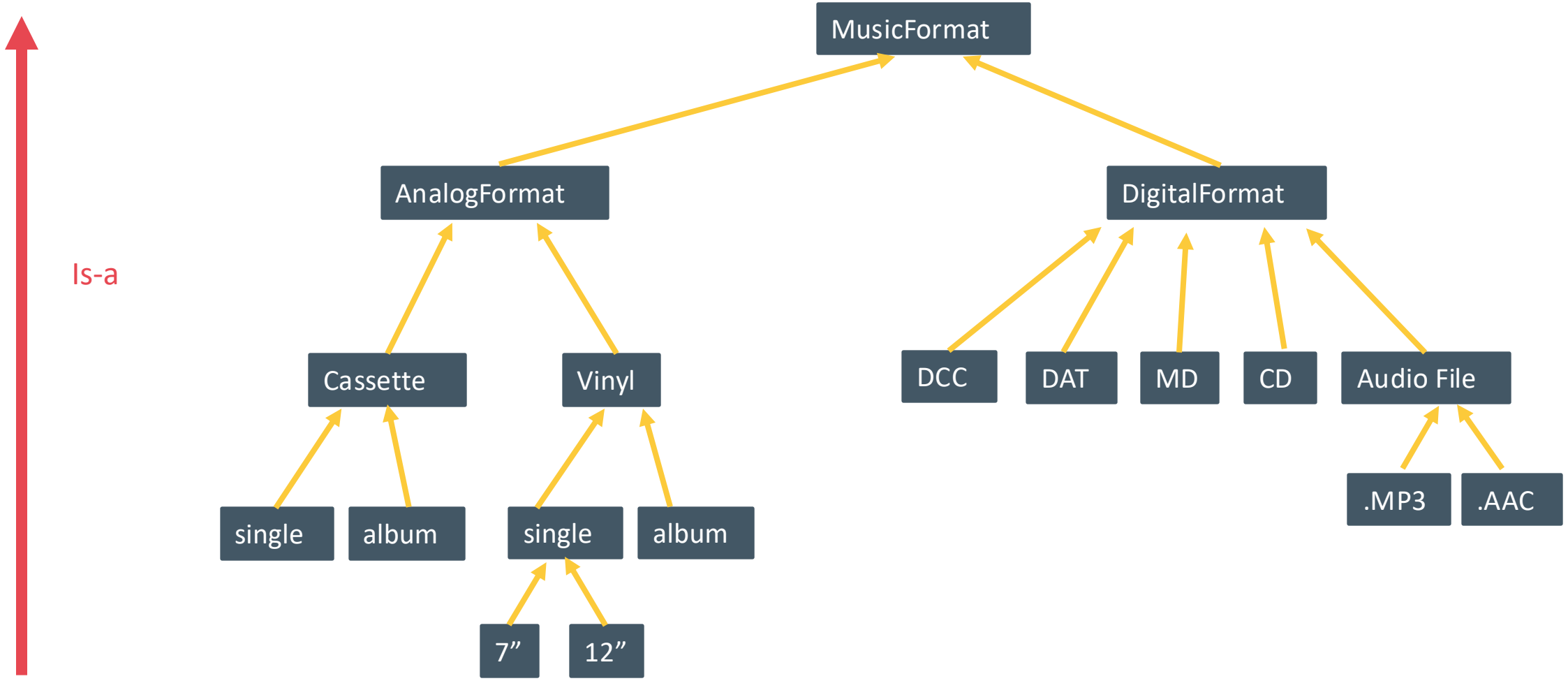
Inheritance Hierarchies

Inheritance hierarchies



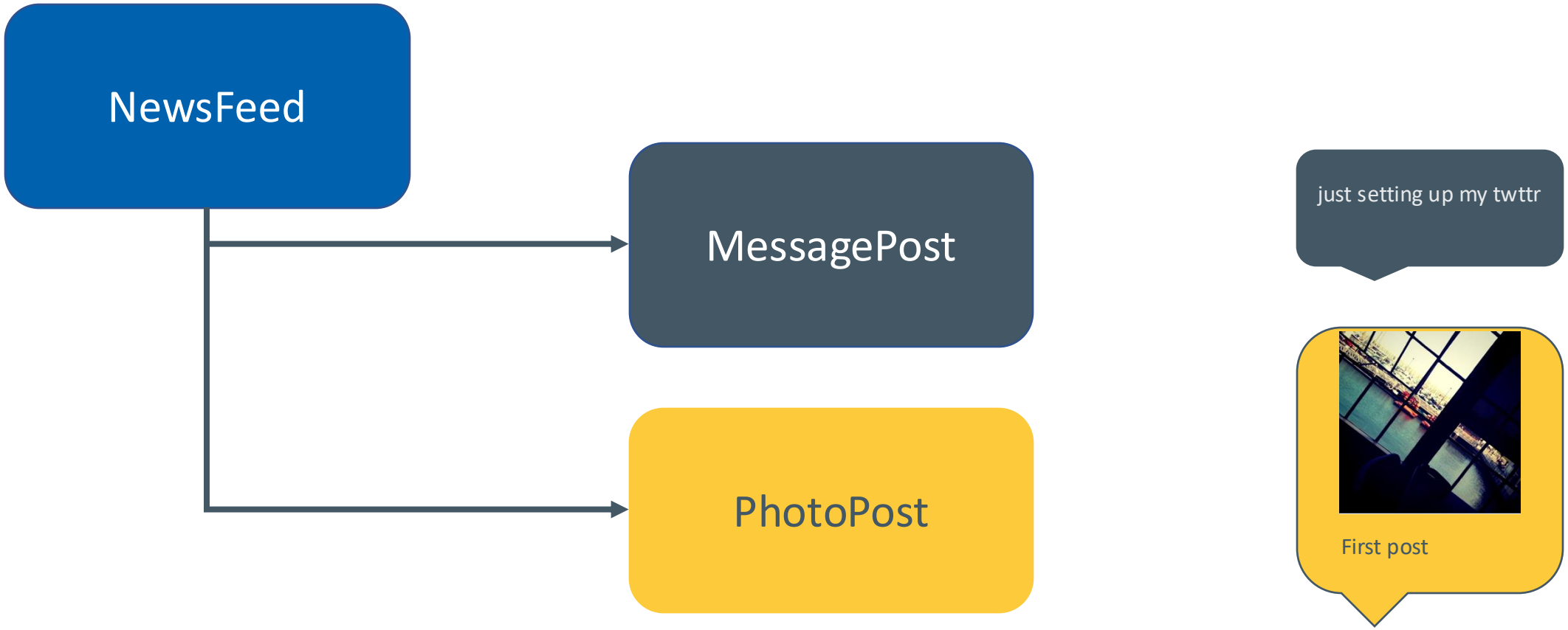
Inheritance hierarchies

Question: Create an inheritance hierarchy for something that you can relate to



Social Network v5

Social Network v4 Recap – Class Diagram

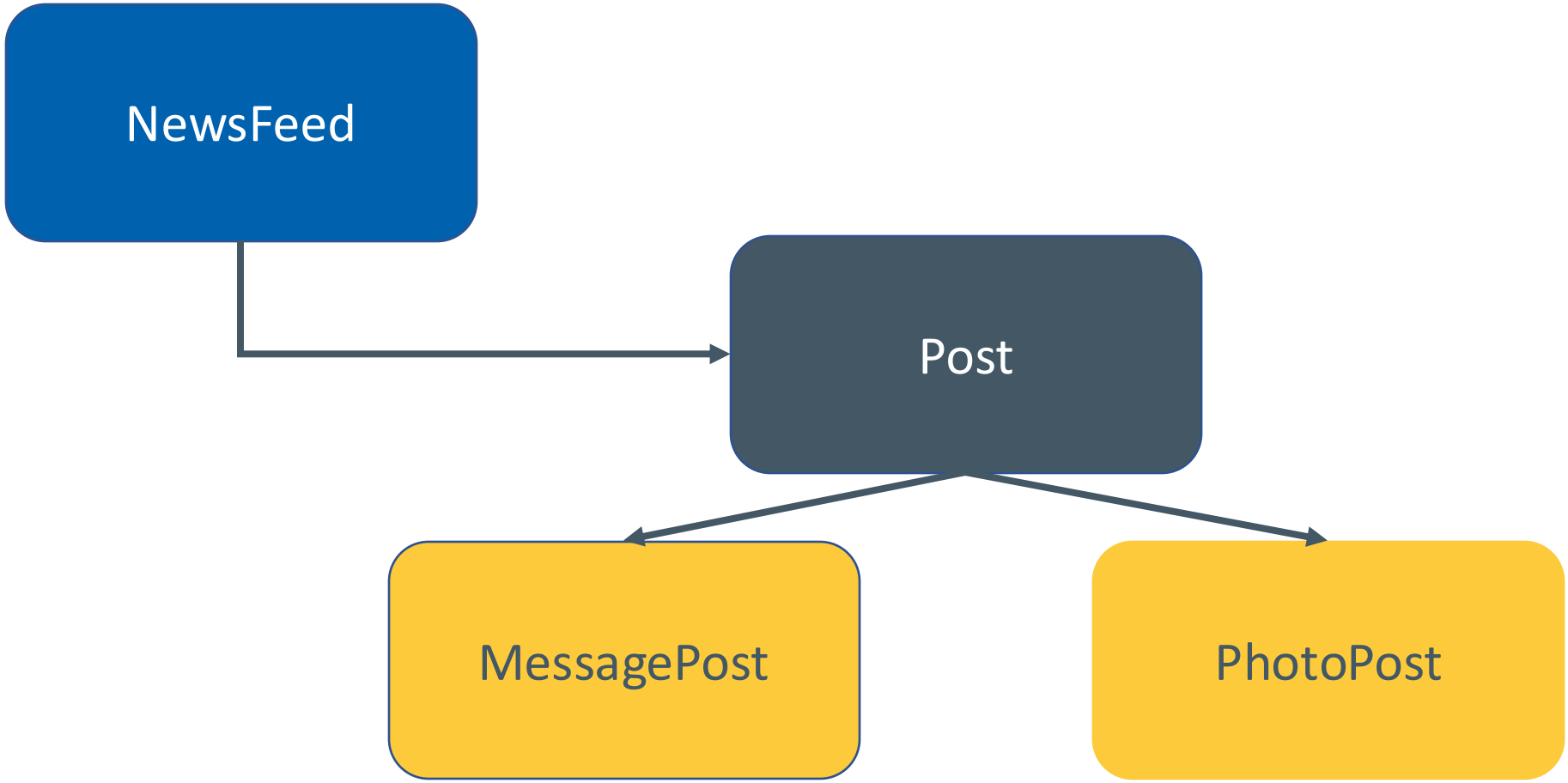


Critique of Social Network v4

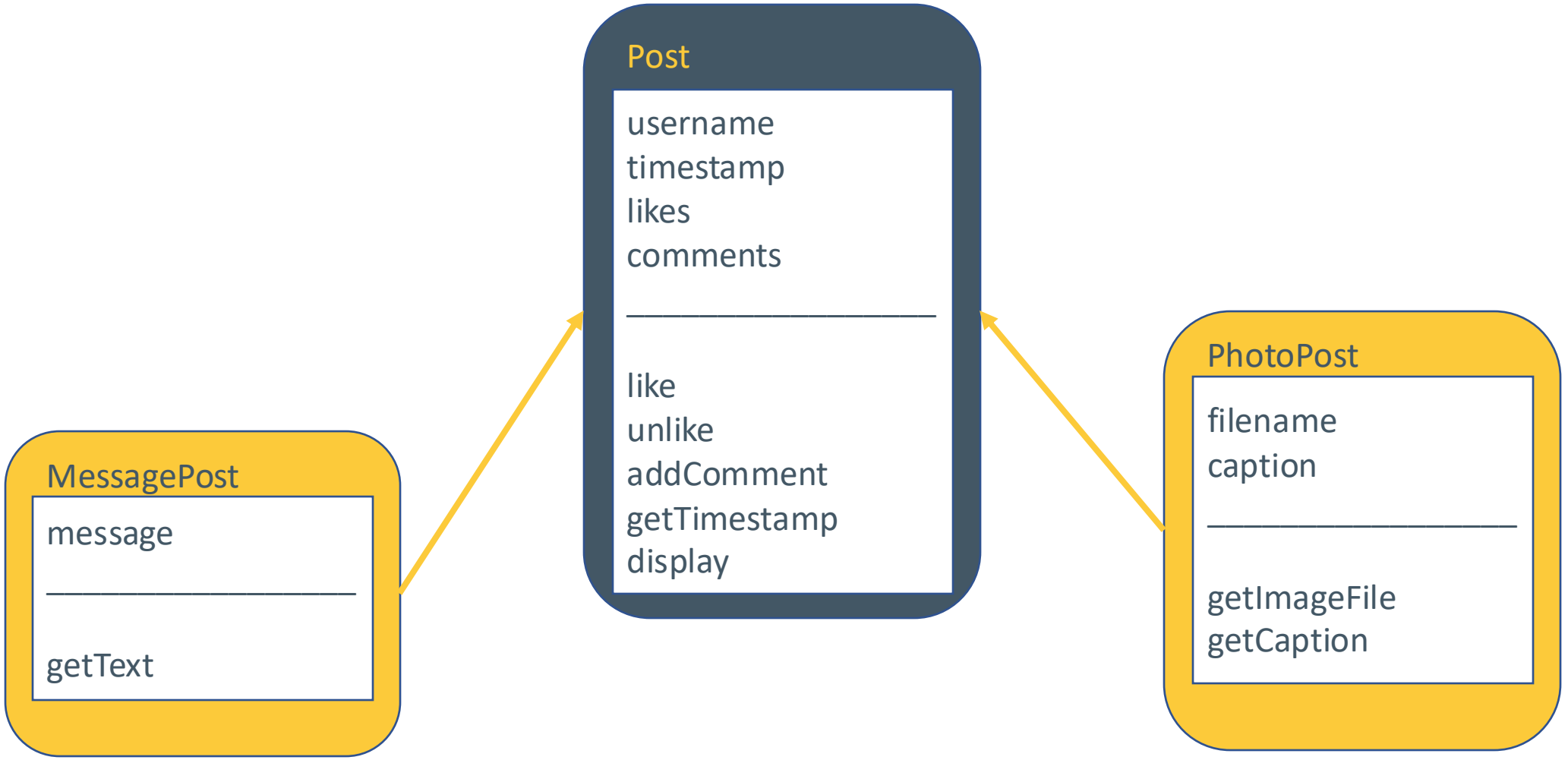
- Code duplication:
  - MessagePost and PhotoPost classes *very similar (large parts are identical)*
  - makes **maintenance** difficult/more work
  - introduces **danger of bugs** through incorrect maintenance

- Code duplication in **NewsFeed class** as well.

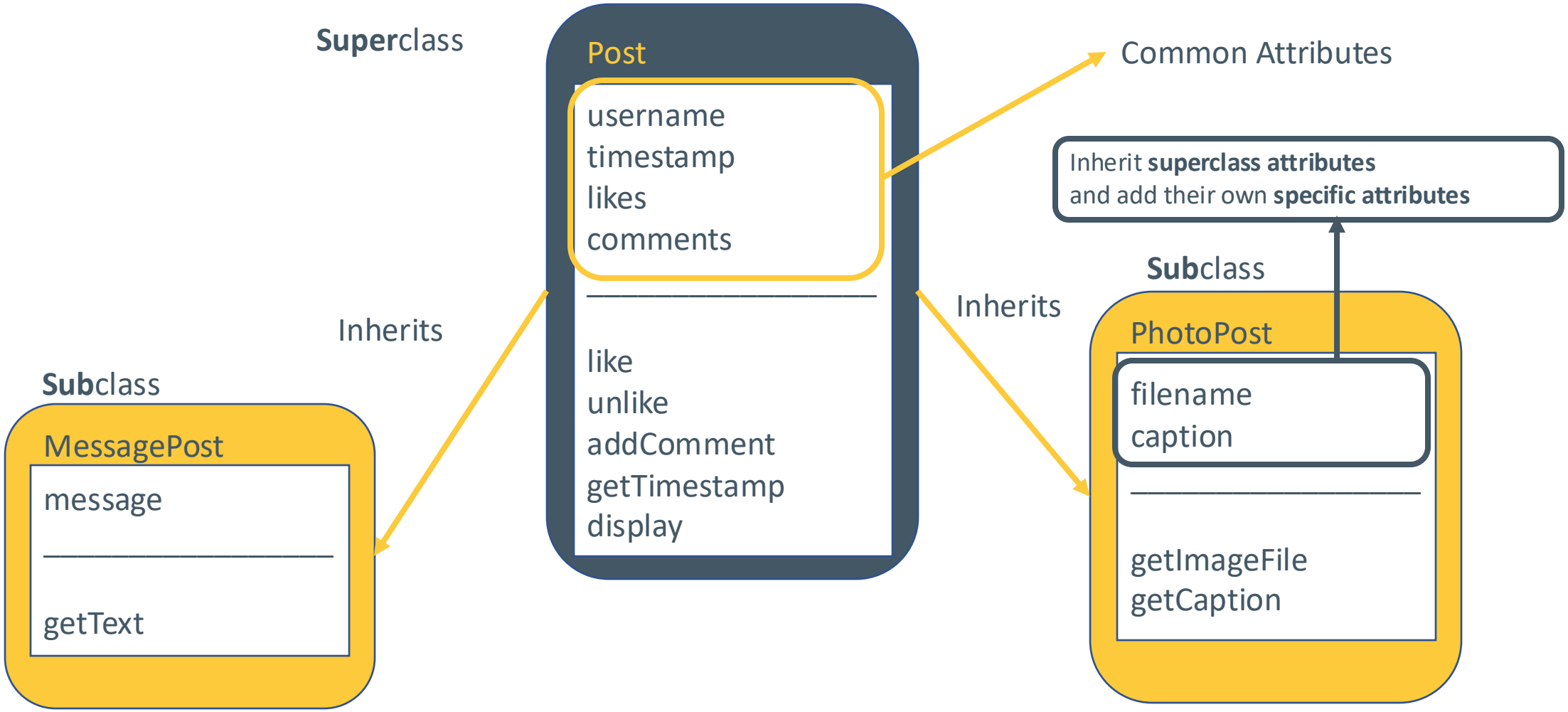
Social Network v5 Class Diagram



Social Network v5 – Classes



Social Network v5 – Classes

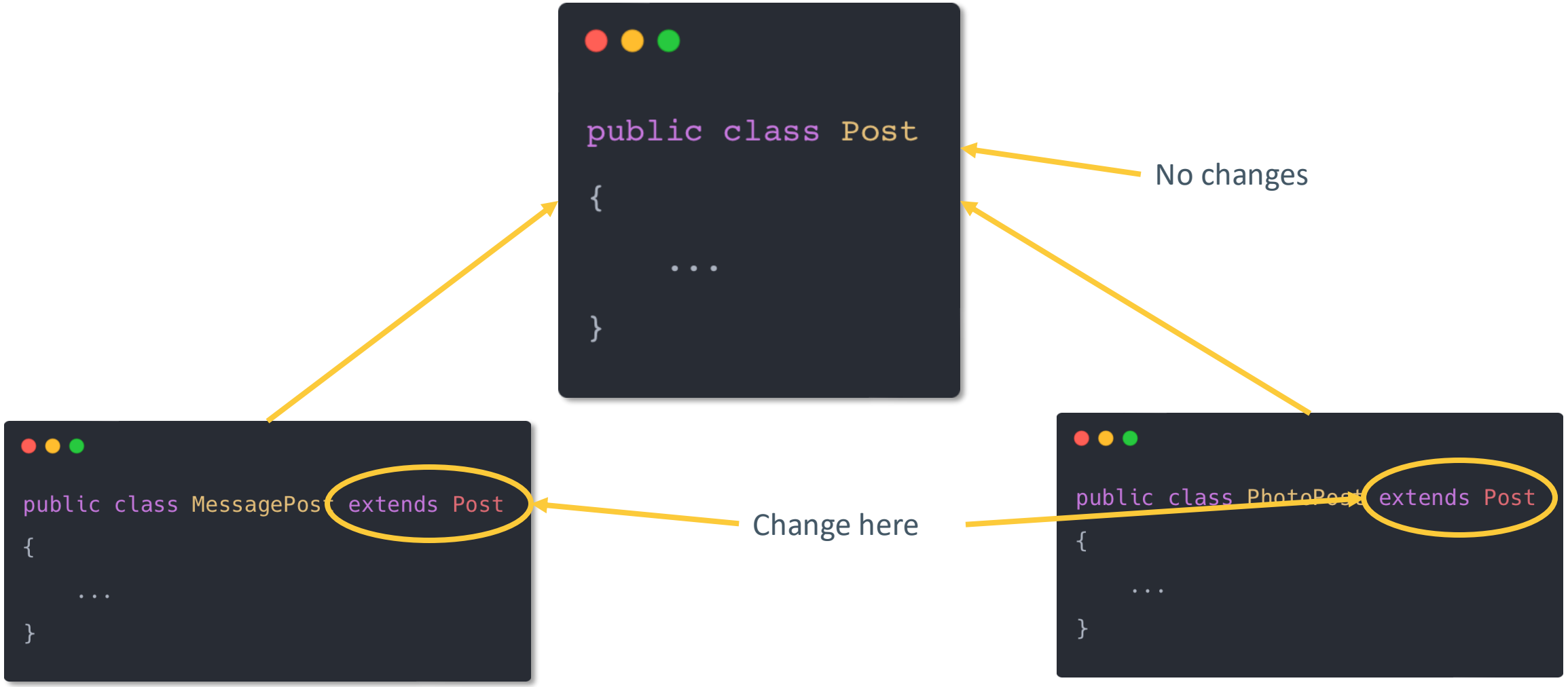


Social Network v5 – Inheritance summary

- define one **superclass**
  - **Post**
- define **subclasses** for
  - **MessagePost**
  - **PhotoPost**
- the **super**class
  - defines common attributes (via fields)
- the **sub**classes
  - **inherit** the superclass attributes (fields)
  - add other specific attributes (fields)

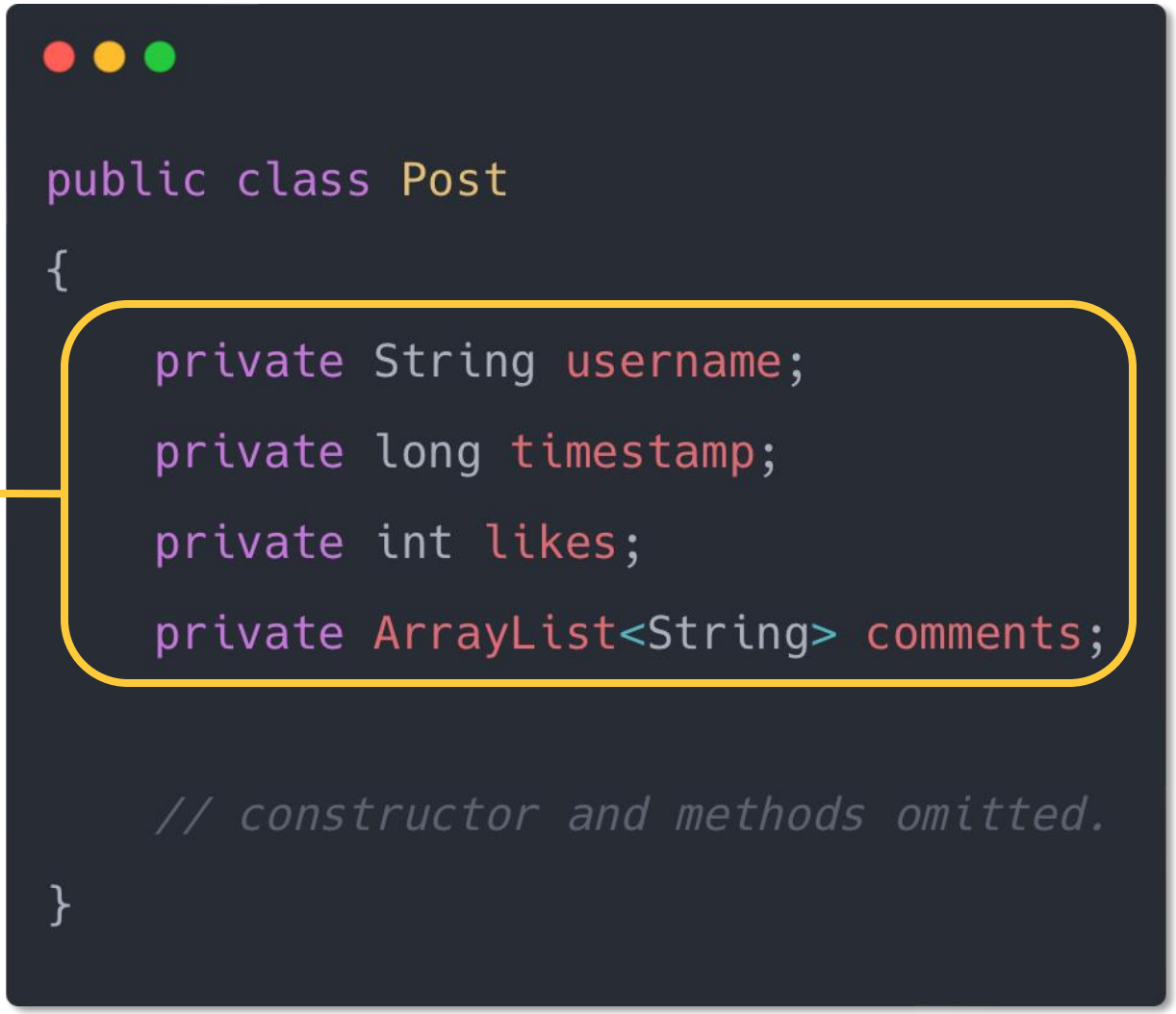
Super and Sub classes – code

Inheritance in Java



Superclass

Common Fields



```
public class Post
{
    private String username;
    private long timestamp;
    private int likes;
    private ArrayList<String> comments;

    // constructor and methods omitted.
}
```

Subclasses

```
public class PhotoPost extends Post
{
    private String filename;
    private String caption;

    // constructor and methods omitted.
}
```

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

    // constructor and methods omitted.
}
```

Subclass fields are unique to that subclass

Subclass objects inherit all fields from the superclass

Inheritance & Constructors – superclass

Nothing unusual in the superclass

```
public class Post
{
    private String username;
    private long timestamp;
    private int likes;
    private ArrayList<String> comments;

    /**
     * Initialise the fields of the post.
     */
    public Post(String author)
    {
        username = author;
        timestamp = System.currentTimeMillis();
        likes = 0;
        comments = new ArrayList<String>();
    }

    // methods omitted
}
```

Inheritance & Constructors – subclass

- Must extend the superclass
- Must call the superclass constructor

```
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
}
```

Superclass constructor call

- Subclass constructors must always contain a **super** call.

- If none is written, the compiler inserts one (without parameters)
  - works only, if the superclass has a constructor without parameters

- **super** call must be the first statement in the subclass constructor.

Questions