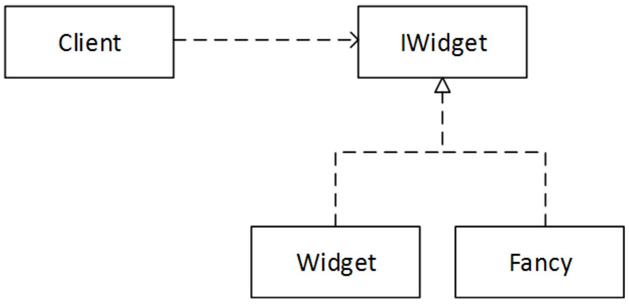
Java Generics

Comp 303

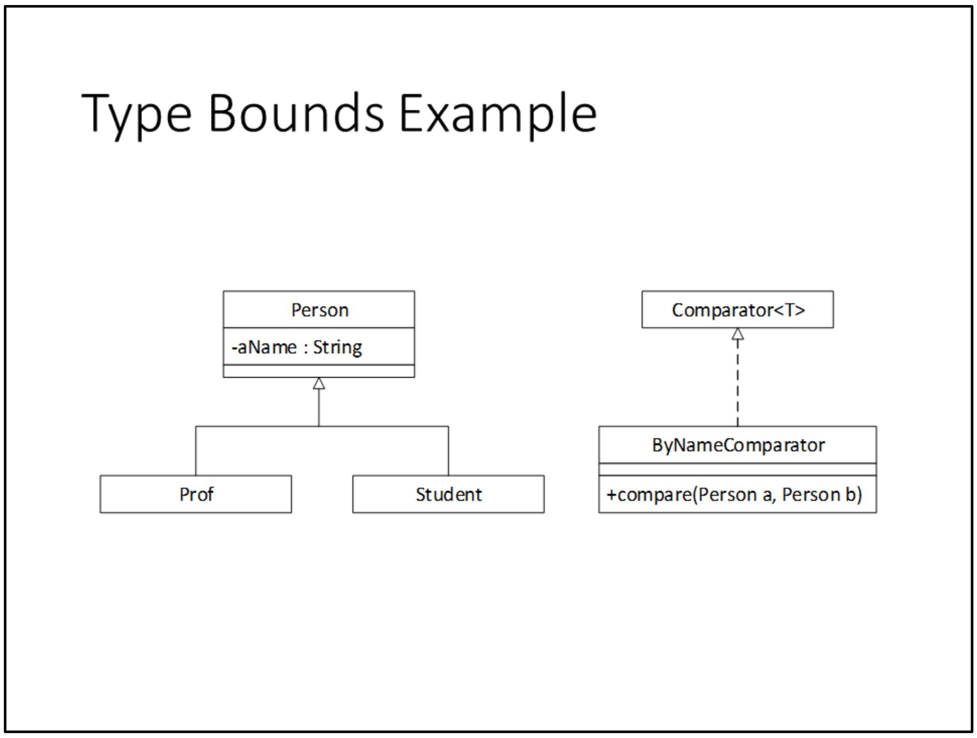
Java Generics

```
ArrayList<String> ar = new ArrayList<String>();
   ar.add( "Hey!" );
   ar.add( new Double(0) );
```

Subtype Polymorphism



Every Java class is a subclass of `Object`.



Type Bounds Example

What is the class declaration of ByNameComparator?

```
public class ByNameComparator implements Comparator<Person> {
....
}
```

The super-type of the ByNameComparator is

```
                    Comparator<Person> = new ByNameComparator();
```

How can we implement Arrays.sort?

```
                    public static void sort(Object[] a, Comparator c) { … }
```

With Java generics this looks like

```
                    public static <T> void sort(T[] a, Comparator<T> c) { … }
```

We could use this in the following way:

```
                    Student[] s = ...;
                    Comparator<Student> c = new Comparator<Student>() {
                                        @Override
                                        public int compare(Student arg0, Student arg1) {
                                                            // TODO Auto-generated method stub
                                                            return 0;
                                        }
                    };
                    Arrays.sort(s, c);
```

Can we use the ByNameComparator with our sort function for an array of Students? No! But we should be able to.
We can use a type bounds in the opposite direction.
public static <T> **void** sort(T[] a, Comparator<E **super** T> c) { … }
Or:
public static <T> **void** sort(T[] a, Comparator<? **super** T> c) { … }
With this declaration we can use the ByNameComparator on an array Student[].