**Using Wildcards With Generics**

Wildcards are represented by the question mark *?* in Java, and we use them to refer to an unknown type. Wildcards are particularly useful with generics and can be used as a parameter type.

But first, there is an important note to consider. **We know that *Object* is the supertype of all Java classes.** However, a collection of *Object* is not the supertype of any collection.

For example, a *List<Object>* is not the supertype of *List<String>*, and assigning a variable of type *List<Object>* to a variable of type *List<String>* will cause a compiler error. This is to prevent possible conflicts that can happen if we add heterogeneous types to the same collection.

The same rule applies to any collection of a type and its subtypes.

Consider this example:

**public** **static** **void** **paintAllBuildings**(List<Building> buildings) {

buildings.forEach(Building::paint);

}

If we imagine a subtype of *Building*, such as a *House*, we can't use this method with a list of *House*, even though *House* is a subtype of *Building*.

If we need to use this method with type *Building* and all its subtypes, the bounded wildcard can do the magic:

**public** **static** **void** **paintAllBuildings**(List<? extends Building> buildings) {

...

}

Now this method will work with type *Building* and all its subtypes. This is called an upper-bounded wildcard, where type *Building* is the upper bound.

We can also specify wildcards with a lower bound, where the unknown type has to be a supertype of the specified type. Lower bounds can be specified using the *super* keyword followed by the specific type. For example, *<? super T>* means unknown type that is a superclass of *T* (= T and all its parents).