**A factory** is **simply a wrapper function around a constructor** (possibly one in a different class). The key difference is that a factory method pattern **requires the entire object to be built in a single method call**, with all the parameters passed in on a single line. The final object will be returned.

**A builder pattern**, on the other hand, **is in essence a wrapper object around all the possible parameters** we might want to pass into a constructor invocation. This **allows us to use setter methods** **to slowly build up our parameter list**. One additional method on a builder class is a **build() method**, which simply **passes the builder object into the desired constructor**, and **returns the result**.

In static languages like Java, this becomes more important when we have **more than a handful of (potentially optional) parameters**, as it **avoids the requirement to have telescopic constructors** for all the possible combinations of parameters.

