1. Factories (Factory method and Abstract Factory)

Motivation:

1. Object creation logic becomes too convoluted
2. Constructor is not descriptive
   1. Name mandated by name of containing type
   2. Cannot overload with same sets of arguments with different names
   3. Can turn into overloading hell
3. Wholesale object creations (non-piecewise, unlike Builder) can be outsourced to
   1. A separate function (Factory Method)
   2. That may exist in a separate class (Factory)
   3. Can create hierarchy of factories with Abstract Factory

Factory: A component responsible solely for the wholesale (not piecewise) creation of objects. (One call for creation)

2) Factory Method

Something like this is not allowed in java:

public class Point{

Point(double a){}

Point(double b){}

} //Example of cartesion and polar coordinate points

 So we can use the Factory method.

1. We will create static functions for wanted constructors. And make the original constructor private.
2. Example code written with points.

3) Factory

Put static classes created in the previous example into a separate class. For this you need to make the constructor public again which you may not want.

Second way to do it is to put the factory class as a static class inside the objects class and still make the constructor private.

4) Abstract Factory

Example written with hot drink and hot drink interfaces.

5) Summary

1. A factory method is a static method that creates objects.
2. A factory can take care of abject creation.
3. A factory can be external or reside inside the object as an inner class.
4. Hierarchies of factories can be used to create related objects.