# Abstract factory Design Pattern

A good example in the Java space is the JAXP API. JAXP stands for Java API for xml parsing. Using this API we can read or write and update the elements in an xml file the key class in this API is the document class that represents an xml document in memory to create a document class.

We use that document builder. So this document builder is a factory class and there is one more class document. Build a factory which is responsible for creating the document builder itself. So the document builder factory is an abstract factory because it is a factory of factories and the use case you are going to work on to give you

Another example is a DAO factory. DAO stands for data access object. We’ll learn more about it in sections later on. It simply is a class that can read write create update data we can have different types of DAO'sDAO's that deal with xml data and DAO's that deal with DB data and within xml we can have employee information department information.

Similarly within that database we can have employ information and department information. So you can see that we can have a factory to deal with these separate DOA's we can have a DB DAO factory that can give us one of these classes here when our application needs them and we can have a xml DAO factory which can give one of the classes here.

Now to get one of these factories themselves these factories will first implement a DAO abstract factory or they will extend DAO abstract factory class and we will have a producer which is responsible for creating one of these factories so abstract factory is a factory off factories.

It simply creates the factory we need. When we have multiple factories we see in our application.

# UML

