Hay tres categorías de patrones de diseño:

* Creacionales: determinan como crear objetos
* Estructurales: Cómo usar objetos entre si mediante composición
* De comportamiento: cómo comunicar objetos sin componerlos directamente

Creacionales

-Factory

-Builder

-Singleton

-Prototype

Estructurales

-Adapter

-Bridge

-Composite

-Decorator

-Facade

-Proxy

De comportamiento

-Interprete

-Cadena de responsabilidad

-Comando

-Iterador

-Memento

-observer

Patron Factory: crea una clase factory que crea objetos concretos mediante un selector. El caller no conoce las clases concretas

Interface Dog{

Void bark();

}

Class Rottweiller implements Dog {

Void bark(){

}

}

Class poodle implements Dog{

Void bark (){

}

}

Class Dog\_Factory{

Public **static** Dog getDog(DogTyped){

If (d==DogType.BIG ){

Return **new** Rottweiller();

}**else** **if** (d==Dogtype.SMALL){

Return **new** Poodle();

}**else**{

Throw **new** Exception(“Unknow Dog Type”);

}

}

**enum** DogType{

SMALL;

BIG;

}

}

Facade: Crea una clase que funciona como Fachada para un subsistema complejo. El caller interactua con el Facade y no con el subsistema directamente