THE FACTORY PATTERN

REFLECTION?

IN THE EXAMPLE ABOVE, REFLECTION PLAYED AN IMPORTANT ROLE, AND THIS IS WORTH TALKING ABOUT A BIT MORE

REFLECTION IS A WAY TO INVOKE METHODS OF OBJECTS ON THE FLY (AT RUN-TIME)

REFLECTION IS AVAILABLE IN MOST IMPORTANT LANGUAGES NOW (JAVA, C# ETC), BUT..

REFLECTION IS SLOW AND COMPLICATED

WHAT IF WE WANTED TO USE THE FACTORY PATTERN WITHOUT REFLECTION?

(ASIDE: WE HAVE THESE 2 CLASSES DERIVE FROM AN ABSTRACT BASE CLASS AND NOT AN INTERFACE BECAUSE THEY SHARE IMPLEMENTATION, NOT BEHAVIOR)

WE WOULD HAVE TO EDIT THE CODE AT SWITCHOVER - NO OPTION NOW

BUT WE WOULD STILL DO THIS SWITCHOVER IN A RELATIVELY SMART WAY

HAVE 2 FACTORY CLASSES -

OracleDatabaseFactory

MSSQLDatabaseFactory

EACH OF THESE FACTORIES ONLY RETURNS DATABASE OBJECTS OF A SPECIFIC TYPE

```
public abstract class DatabaseFactory {

     abstract IDatabase getDatabase();

     private String readFromConfig(String key) {
         // the config file has key-value pairs,
         // return the value corresponding to the key specified
     }
}

public MSSQLServerDatabaseFactory extends DatabaseFactory {

     public IDatabase getDatabase() {
        return new MSSQLServerDatabase();
     }
}


public OracleDatabaseFactory extends DatabaseFactory {

     public IDatabase getDatabase() {
        return new MSSQLServerDatabase();
     }
```

1. CREATE THE ABSTRACT FACTORY OBJECT - LEAVE OUT THE ACTUAL GETDATABASE METHOD

BTW NOTE HOW THE GETDATABASE METHOD IS NO LONGER STATIC

2. CREATE THE 2 ACTUAL FACTORY OBJECTS

```
    public IDatabase getDatabase() {
        return new MSSQLServerDatabase();
    }
}

public OracleDatabaseFactory extends DatabaseFactory {

    public IDatabase getDatabase() {
        return new MSSQLServerDatabase();
    }
}

AbstractDatabaseFactory databaseFactory = new MSSQLServerDatabaseFactory()
IDatabase database = databaseFactory.getDatabase()
```

2. CREATE THE 2 ACTUAL FACTORY OBJECTS

3. INSTANTIATE THE CORRECT FACTORY, AND THEN USE THAT TO GET THE ACTUAL DATABASE OBJECT

NOW, ONCE WE SWITCH FROM MS-SQL SERVER TO ORACLE, WE WOULD NEED TO EDIT THE LAST BIT SO THAT WE NOW USE AN ORACLEFACTORY OBJECT

THIS IS A VARIATION OF THE FACTORY PATTERN, THAT IS CALLED THE

ABSTRACT FACTORY PATTERN

USE ABSTRACT FACTORY TO CREATE FAMILIES OF RELATED CLASSES

BTW - THERE IS ONE SOLUTION TO THIS PROBLEM THAT YOU SHOULD NEVER EVER EVER USE

THAT SOLUTION IS TO CREATE THE DATABASE OBJECT USING CODE LIKE THIS

```
public class DatabaseFactory {

    public static IDatabase getDatabase(String databaseType) {

        if(databaseType == "MS-SQL"){
          return new MSSQLDatabase();
        }

        if(databaseType == "Oracle"){
          return new OracleDatabase();
        }

        if(databaseType == "MySQL"){
          return new MySQLDatabase();
```

WRITING CODE LIKE THIS IS A FAUX PAS,
UNACCEPTABLE - ITS LIKE PICKING YOUR
NOSE IN PUBLIC

WHY? BECAUSE EACH TIME A NEW
DATABASE TYPE NEEDS TO BE SUPPORTED,
THIS CLASS CODE NEED TO BE UPDATED

AND SINCE CIOS CHANGE PRETTY
OFTEN, THIS CLASS WILL BE GETTING
A LOT OF REWRITES

THE OPEN-CLOSE PRINCIPLE

"CODE SHOULD BE OPEN FOR EXTENSION
BUT CLOSED FOR MODIFICATION"