THE FACTORY PATTERN

LET'S SAY WE ARE WRITING AN APPLICATION THAT PULLS DATA FROM A DATABASE AND DOES STUFF WITH IT

(PRETTY COMMON USE-CASE)

OUR COMPANY CURRENTLY USES MS-SQL SERVER, BUT IS IN THE PROCESS OF MOVING TO ORACLE

(PRETTY COMMON ALSO, HAPPENS EACH TIME A NEW CIO TAKES OVER)

HOW DO YOU MINIMIZE THE PAIN ASSOCIATED WITH THIS CHANGE IN DATABASE SERVERS?

USE THE

FACTORY PATTERN

THE DUMB WAY:

PRE-SWITCHOVER

```
IDatabase database = new MSSQLDatabase()
```

POST-SWITCHOVER

```
IDatabase database = new OracleDatabase()
```

WHY IS THIS NOT A GREAT WAY OF DOING IT? BECAUSE THE SWITCHOVER INVOLVES MODIFYING CODE, AND RE-COMPILING, AND RE-RELEASING INTO PRODUCTION

IN PRODUCTION SYSTEMS THIS CAN BE AN ONEROUS, COMPLICATED AND ERROR-PRONE PROCESS

THE SMART WAY

```
IDatabase database = DatabaseFactory.getDatabase()
```

YOU SIMPLY ASK A "FACTORY OBJECT" TO GIVE YOU THE CORRECT TYPE OF DATABASE OBJECT, AND IT KNOWS HOW.

WHAT IS THIS FACTORY OBJECT, AND HOW DOES IT KNOW WHAT TYPE OF OBJECT TO INSTANTIATE?

THE FACTORY OBJECT READS IN THE NAME OF THE CLASS TO INSTANTIATE FROM A CONFIG FILE, AND USES REFLECTION TO INSTANTIATE THE CORRECT OBJECT

(ANY SERIOUS PRODUCTION SYSTEM WILL HAVE AT LEAST 1 CONFIG FILE)

A CONFIG FILE IS SIMPLY A TEXT FILE USED TO SPECIFY INPUTS THAT THE APPLICATION USES AT RUN-TIME

THE AWESOME THING ABOUT CONFIG FILE IS - CHANGING A CONFIG FILE DOES NOT NEED CODE TO BE CHANGED AND RE-RELEASED