**Creational Patterns:** provide object creation mechanisms that increase flexibility and reuse of existing code.

**FACTORY METHOD**

This is a creational design pattern that provides an interface for creating objects in a superclass, but allows subclasses to alter the type of objects that will be created.

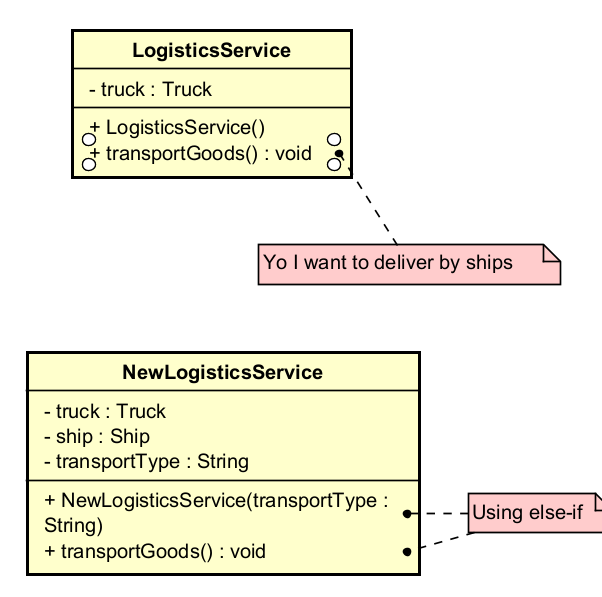
Example:

Imagine buidling a delivery app, at first, it only supports truck. Everything in code is designed around trucks, but then, shipping companies want in, and later, maybe even planes.

If code is tightly coupled to trucks, adding ships means rewriting a lot of things. Later, if adding planes, we have to do it all over again. That’s messy and hard to maintain.

What do you mean ‘adding a new transport means rewriting a lot of things ?’.

Lets take a look of an example of the problem.



Problem: Need if-else conditions everytime adding a new transportation. If you add Plane, you will have to modify all these if-else conditions again. This violates the OPEN/CLOSED PRINCIPLE means that you add new features by changing existing code.

SOLUTION:

