## What is the factory design pattern and why should I use it?

The short definition of the factory pattern is delegating the instantiation of concrete classes to a factory class. Like many design patterns, the best way to understand it is to see an example – which is what we’re going to do today.

The big question, though, is why should I use the factory pattern? The technical answer is that it, when properly applied, aids in:

* Separation of Concerns
* Loose Coupling

I know what you’re thinking. Great, more technical terms you probably need to see an example to understand. However, from a functional perspective, this does four things for our code:

* Makes our code easier to read.
* Makes our code easier to maintain.
* Makes our code easier to test.
* Makes our code easier to scale.

For today’s example, let’s say that we need to write an API that reads and writes files into some sort of accessible, long-term storage location. In this example, we’ll have three locations:

* Local File System
* Amazon S3 Storage
* Google Cloud Storage Buckets

The first things we’re going to look at are the FileStorageConfig class, the IFileStorageService interface, and the concrete, File Storage Service classes.

* Config could be loaded from config file, database, API object, etc.
* None of the methods in the concrete classes have been implemented!

BadFilesController:

* It’s not bad because it wouldn’t work – it would, if the concrete classes were implemented.
* Using a version of the factory design pattern for logging.
* Tightly coupled to the concrete File Storage Service classes.
* Impossible to unit test without relying on the concrete classes and the implemented methods.
* Functionally, this controller cannot be properly unit tested without refactoring the controller code and/or the concrete classes.

File Storage Factory:

* Delegation of the creation of any concrete, file storage service class only exists in this factory – regardless of how many controllers, projects, etc., need to utilize a file storage service.

FilesController:

* Properly implements the factory design pattern using the file storage factory.
* Even though the concrete classes have no implemented methods, this entire controller can be – and currently is – fully unit tested with 100% code coverage.