Dependency Injection

This course is not about dependency injection...

... but clean code is impossible without it!

The Two Faces of Dependency Injection

The act of providing (injecting) the required dependencies using one of the fundamental techniques: constructor, method or field injection

Simple

Dependency Injection Architectural Pattern

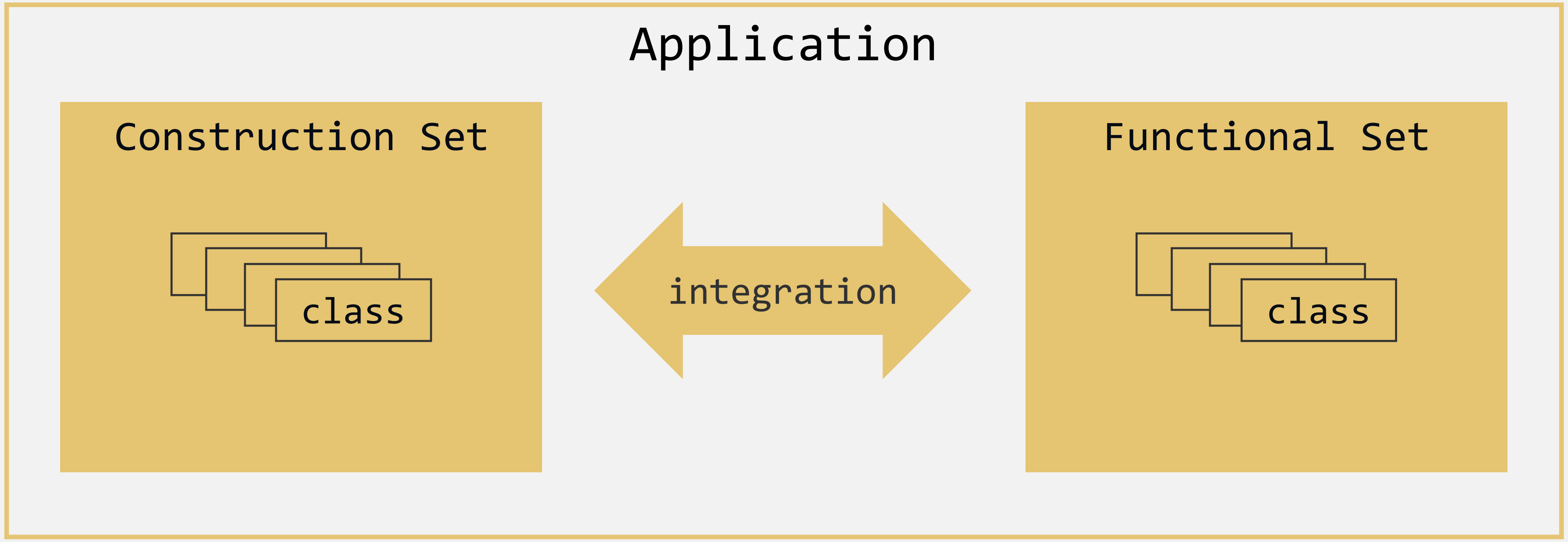
Very complicated

Dependency injection architectural pattern (DIAP):

The main characteristic of DIAP is **segregation of application’s logic into two sets of classes**:

> **Functional set:**
> contains classes that encapsulate core application’s functionality
>
> **Construction set:**
> contains classes that resolve dependencies and instantiate objects from functional set

Functional and construction sets must be **disjoint**.



Segregation of classes into Construction and Functional sets is manifestation of **Separation of Concerns**

Relationship between fundamental dependency injection techniques and DIAP:

Different levels of abstraction (class vs application)

Fundamental techniques – class level Single Responsibility Principle

DIAP – application level Separation of Concerns

Implementations of DIAP use fundamental techniques under the hood

My Goals With Dependency Injection in This Course

Explain the benefits of combining MVC with dependency injection

Demonstrate the intimate relationship between dependency injection and architecture

Show that it doesn’t take much time to implement dependency injection when you know what you’re doing

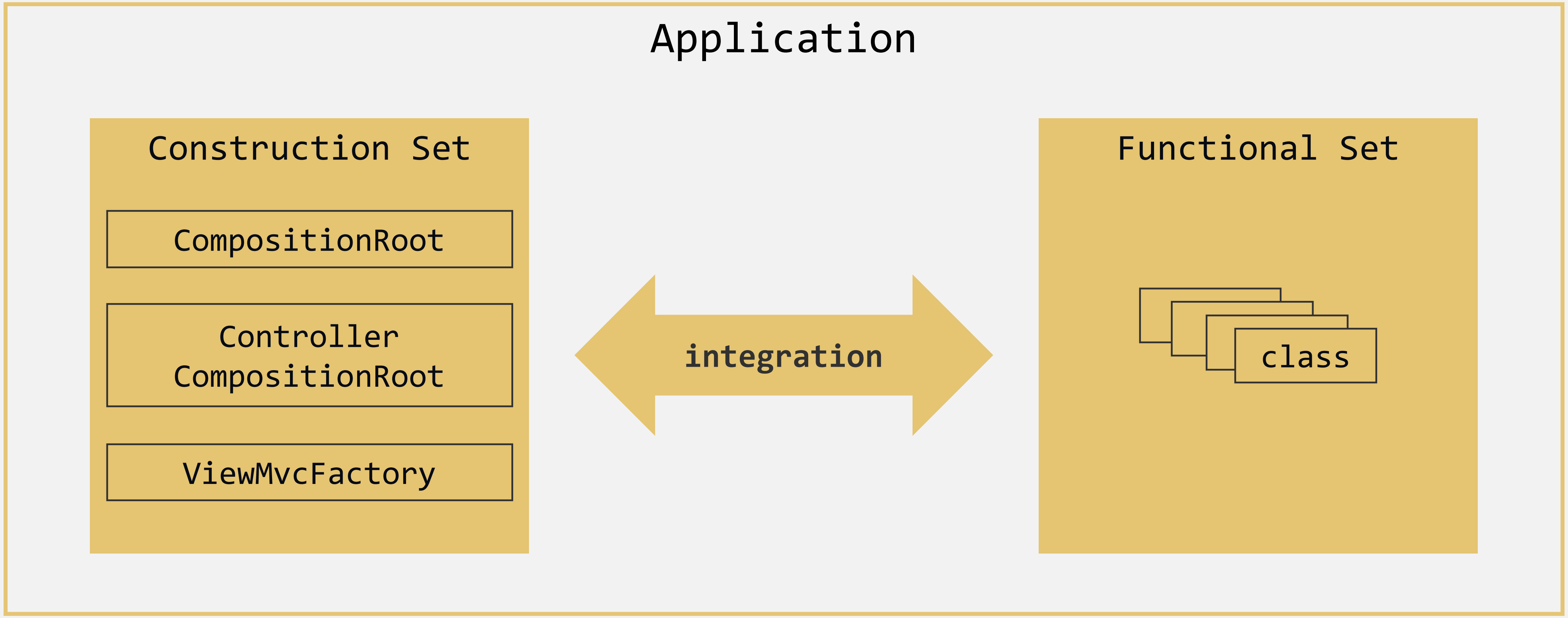
Full theory of dependency injection, as well as dependency injection frameworks are **NOT** in the scope of this course

Pure Dependency Injection

= Poor Man's Dependency Injection ☹

≠ Service Locator!!!

Dependency Injection Summary


Application
Construction Set
CompositionRoot
Controller CompositionRoot
ViewMvcFactory
integration
Functional Set
class

My Goals With Dependency Injection in This Course

Explain the benefits of combining MVC with dependency injection

Demonstrate the intimate relationship between dependency injection and architecture

Show that it doesn’t take much time to implement dependency injection when you know what you’re doing