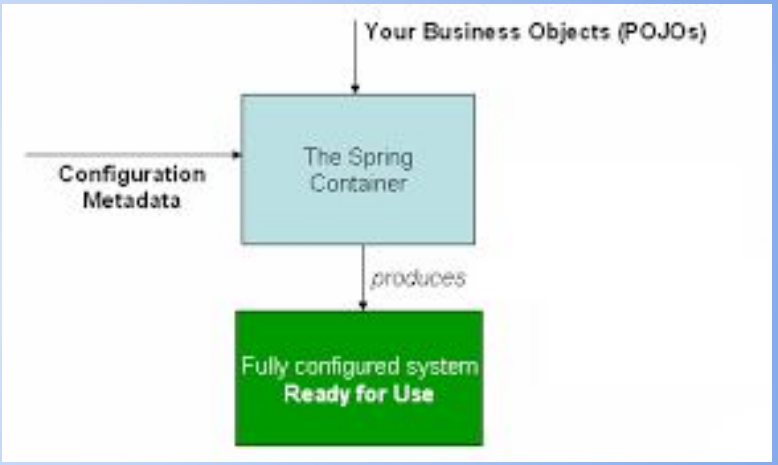
Spring

- Spring is an application framework and inversion of control (also known as dependency injection) container for Java.
- Spring is open source and its core features can be applied to any Java application.
- Spring has begun to become a popular extension to Java because of its functionality for dependency injection.

What is the IoC container

- The inversion of control container (IoC) is the means by which Spring configures and manages Java objects.
- Spring maintains these objects using reflection.
- The primary purpose of the container is to manage object lifecycles.
- Objects created by the container are called manage objects or “beans”.



What is Dependency Injection

- Dependency Injection is the process by which an object supplies the dependencies of another object.
  - A “dependency” is the use of a target objects functions. In other words if one class needs a function from another class we would say it’s a dependency
- In Java, before one can use methods of other classes, they must first create the object of that class (i.e. class A needs to create an instance of class B to use B’s method)
- The responsibility of creating an object along with directly using its dependencies is called dependency injection.
- Connecting an object, or “Injecting” an object is done by the assembler rather than by the objects themselves.

Traditional Way

```
1  public class Store {
2      private Item item;
3
4      public Store() {
5          item = new ItemImpl1();
6      }
7  }
```

Dependency Injection

```
1  public class Store {
2      private Item item;
3      public Store(Item item) {
4          this.item = item;
5      }
6  }
```

Dependency Injection Pros

- Dependency injection provides flexibility since it is configurable.
- It’s easy to switch between different implementations (as we will see in demo)
- Greater modularity of a program
- Reduction of boilerplate code in the application objects, since all work to initialize or set up dependencies is handled by a provider component
- Greater ease in testing a program by isolation a component or moocking its dependencies and allowing a component to communicate.

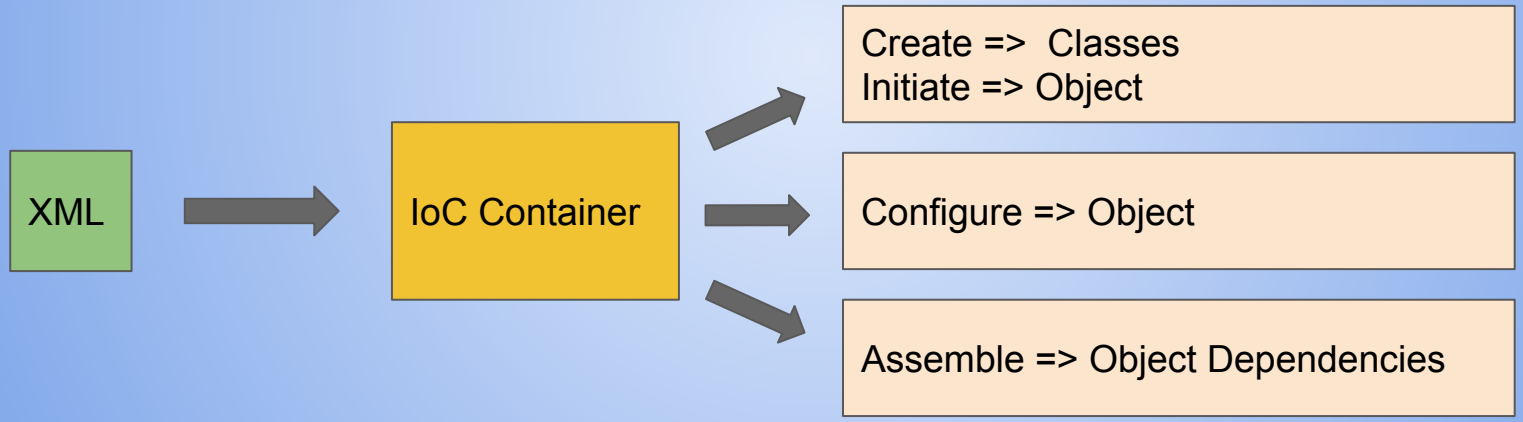
Dependency Injection Cons

- Dependency injection can make code difficult to trace (read) because it separates behavior from construction. This means developers must refer to more files to follow how a system performs.
- Dependency injection frameworks are implemented with reflection or dynamic programming. This can hinder use of IDE automation, such as "find references", "show call hierarchy" and safe refactorings.
- Dependency injection typically requires more upfront development effort since one can not summon into being something right when and where it is needed but must ask that it be injected and then ensure that it has been injected.

IoC Example Approach

Responsibilities of IoC container when working with Objects:

- Instantiating
- Configuring
- Assembling



Connection to Course Content

- Using code as data
  - Objects can be conditionally injected
  - Specify a configuration variable which determines which object should be used
- More efficiently maintainable code
  - No need to specify objection creation boiler plate code
  - Objects are injected in instantiated form