**Inversion of Control**  
  
L’inversione di controllo è un concetto/approccio di esternalizzare la costruzione degli oggetti.  
Quindi invece di andare a creare gli oggetti manualmente, li affideremo a un’altra entità.

Qui entra in gioco il contenitore di spring (spring container) il quale funziona come una fabbrica di oggetti, dandoci o iniettandoci per noi le dipendenze di cui abbiamo bisogno  
  
Quindi possiamo dire che il contenitore di spring ha due importanti funzionalità tra cui:   
  
**- Creare e gestire gli oggetti usando l’inversione di controllo e**

**- Poi anche iniettare le dipendenze degli oggetti facendo uso della dependency injection  
  
Il container spring può essere configurato per permetterci di fare diverse cose, ma come possiamo configurarlo?**Si può ricorrere a tra approcci diversi:  
- Alla configurazione xml → **considerata legacy**  
- Alle annotazioni java  
- Alle configurazioni tramite codice sorgente java  
  
Quindi per fare un esempio molto breve: in una fabbrica viene costruita un auto, per mettere su un auto ha bisogno di vari componenti, quando questi componenti saranno completi, allora verrà messa insieme l’auto, quindi quello che succede e che saranno fatte delle richieste per i vari pezzi dell’auto che devono essere assemblati.   
Appunto possiamo immaginare la nostra applicazione come una fabbrica la quale comunica con il container di spring che crea gli oggetti **(inversion of control)** di cui abbiamo bisogno e poi tramite la **dependency injection** questi saranno iniettati là dove ne abbiamo bisogno  
  
 Dependency injection  
  
**L’iniezione delle dipendenze sfrutta il principio dell’inversione delle dipendenze**, è il caso in cui il client delega a un altro oggetto la responsabilità di fornire le sue dipendenze  
  
**inversione delle dipendenze** → c’è un principio di separazione delle dipendenze tra le classi. che dice, invece di creare oggetti e gestire le loro dipendenze all’interno di una classe, una classe dovrebbe dichiarare le sue dipendenze come astrazioni (interfacce o classi astratte)  
  
Quindi in sitesi il container di spring adesso ha due principali funzioni, ossia quella di creare oggetti applicando l’inversione di controllo e iniettarli applicando l’iniezione delle dipendenze.  
  
**Per quanto riguarda spring, esistono più tipi di iniezione.  
inizione di costruttori e anche di setter  
  
Per l’inizione di dipendenza** viene utilizzato il cablaggio automatico ossia spring cercherà una classe che corrisponda, e può corrispondere per tipo sia una classe che un’interfaccia, spring la inietterà automaticamente. Per far si che una dipendenza venga iniettata bisogna specificare a spring che quella classe è un component tramite la notazione @Component.