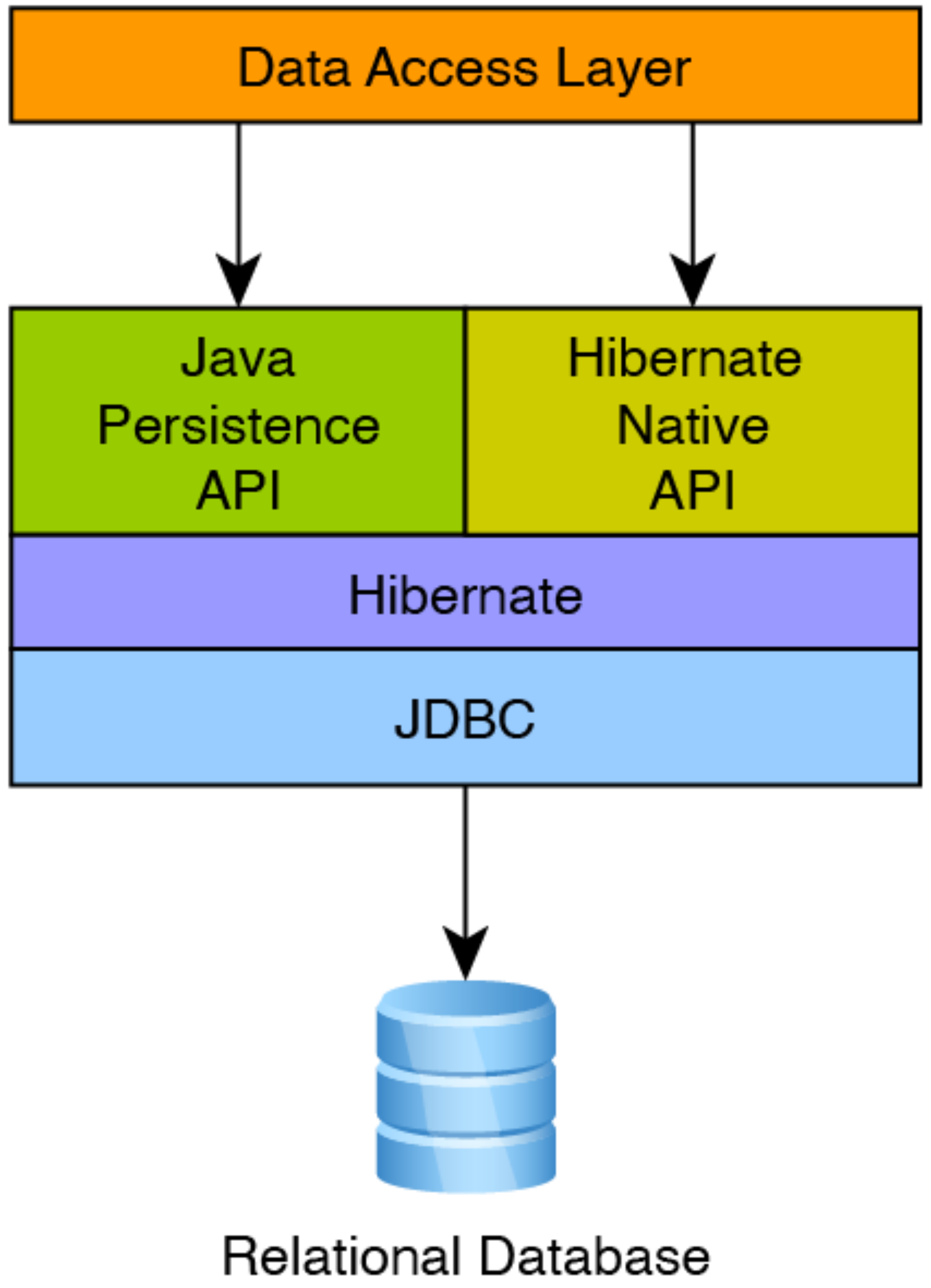
Hibernate & Spring Data JPA

Beginner to Guru

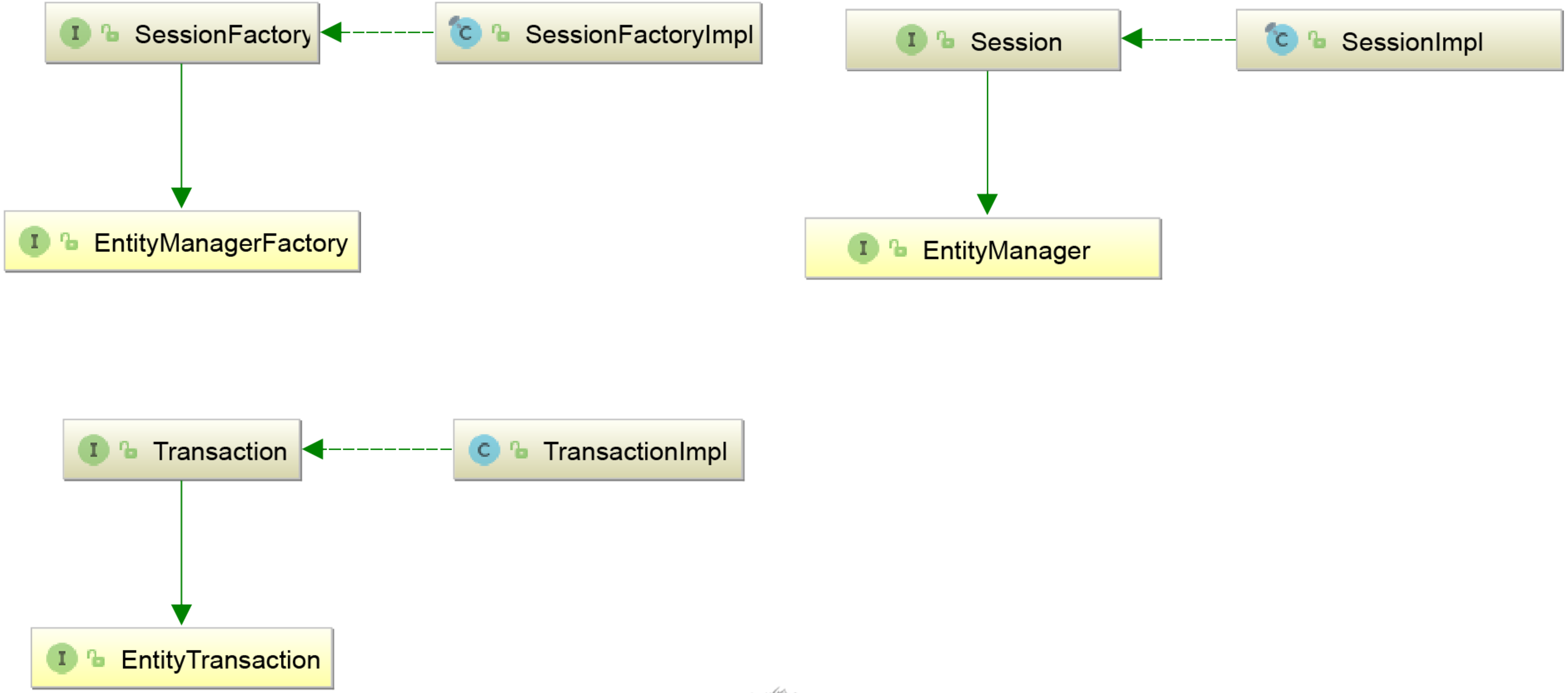
Hibernate Introduction

Hibernate Introduction

- Hibernate is a very popular and mature ORM solution for Java
    - ORM - Object Relational Mapping
- Hibernate also implements the Java JPA specification
    - A lot of the API specification was actually based on Hibernate
- Coding to the Java JPA specification will keep your code independent of Hibernate
- Hibernate also has a native API
    - Coding to this API will make your code dependent on Hibernate



Source: Hibernate Documentation



SessionFactory

SessionFactoryImpl

EntityManagerFactory

Session

SessionImpl

EntityManager

Transaction

TransactionImpl

EntityTransaction

Source: Hibernate Documentation

Hibernate Terms

- Session Factory - Expensive to create, your application should only have one instance
- Entity Manager Factory - JPA Equivalent of Session Factory
- Session - single threaded, short lived object. Cheap to create
  - Session wraps a JDBC connection
- Entity Manager - JPA Equivalent of Session
- Transaction - single threaded, short lived object to define transaction boundaries
- Entity Transaction - JPA equivalent of Transaction

Persistence Context

- Session / Entity Manager - create a context for dealing with persistent data
- Transient - the entity has just been instantiated and is not associated with a persistence context. It has no persistent representation in the database and typically no identifier value has been assigned (unless the assigned generator was used).
- Managed or Persistent - the entity has an associated identifier and is associated with a persistence context. It may or may not physically exist in the database yet.
- Detached - the entity has an associated identifier but is no longer associated with a persistence context (usually because the persistence context was closed or the instance was evicted from the context)
- Removed - the entity has an associated identifier and is associated with a persistence context, however, it is scheduled for removal from the database.

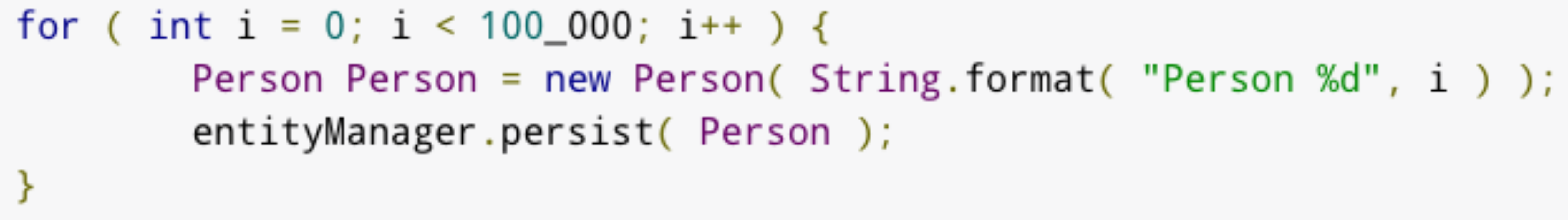
Detached Entities

- Detached Entities - Very common error to see
- Common root cause is working outside of the session scope, or a closed session
- Spring Data JPA by default will do an implicit transaction
  - Meaning you can see this error when accessing entity properties outside of a transaction
  - More on transactions later in the course!

Caching

- Persistence Context / First Level Cache - By default Hibernate will cache entities in the persistence context
  - Changes outside the context might not be seen
  - Very efficient for doing work within the context of a session
- Second Level Cache - Disabled by default. JVM or cluster level cache
  - Recommend to enable on a per entity basis
  - Broad support for popular options such as jCache, Ehcache, and Infinispan

Caching Problems



```
for ( int i = 0; i < 100_000; i++ ) {
        Person Person = new Person( String.format( "Person %d", i ) );
        entityManager.persist( Person );
}
```

- Hibernate would add 100,000 objects to session level cache, possible out of memory
- Long running transaction could deplete transaction pool
- JDBC Batching not enabled by default, each insert is a round trip to the DB.
- flush() and clear() methods can be used to clear session cache

SPRING FRAMEWORK

GURU