## **16. HIBERNATE. Кэш, JPA, логирование, Primary Key**

### First level cache – включен по умолчанию и действует на уровне сессии.

A diagram of a process

Description automatically generated

Session.PresistenceContext – отражает актуальное состояние базы, если в PersistenceContext есть сущность, то запрос в БД не будет отправлен.

1. Session.evict(Entity) – удалить объект из кеша
2. Session.clear() – очистить весь кеш
3. Session.close() – При закрытии сессии кеш удаляется
4. Session.flush() – немедленно обновить БД в соответствии с PersistenceContext

Любые изменения в сущностях ассоциированных с PersistenceContext отразятся в БД при закрытии даже без явного вызова update().

Каждая сессия имеет свой кеш – PersistenceContext.

### Life cycle

A diagram of a process

Description automatically generated

Изначально при объявлении объекта он является Transient , то есть не привязан ни к какому контексту. Если мы получаем объект из сессии или сохраняем его в сессию, тогда он попадает в PersistenceContext. Если мы выполняем удаление объекта, тогда он попадает в Removed. Так уже мы можем удалить (evict), очистить кеш (clear) или закрыть сессию (close), тогда объект становится Detached. Мы можем восстановить объект обратно к контект при помощи saveOrUpdate, update или merge.

1. Session.refresh() – обновляет контекст, получая текущие данные из БД.
2. Session.delete() – удаляет объект из базы данных
3. Session.evict() – удаляет объект из контекста
4. Session.clear() – очищает контекст
5. Session.close() – закрывает сессию
6. Session.merge() – добавляет объект в контекст (или применяет изменения сделанные с объектом к контексту)
7. Session.update() – обновляет объект данными из контекста
8. Session.isDirty() – проверяем если в контексте изменения, которые не были сохранены в базу данных

### Java Persistence API (JPA)

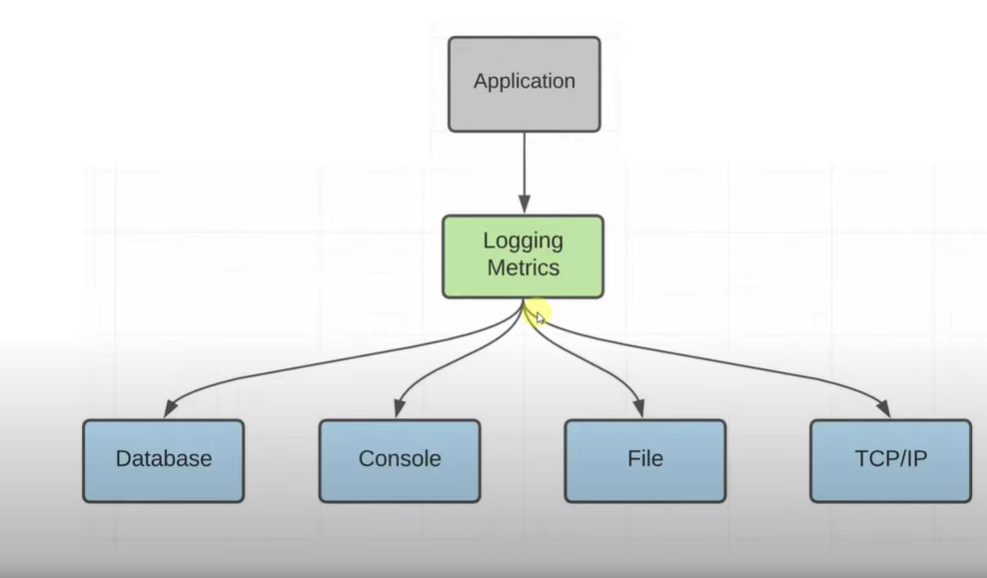
**Java Persistence API** – Спецификация Java, которая предоставляет набор интерфейсов и аннотаций для возможности сохранять в удобном виде Java объекты в базу данных, и наоборот, извлекать информацию из баз данных в виде Java объектов (ORM)

**Hibernate (ORM Framework)** – это одна из самых распространённых JPA реализаций.

Разница между **persist** и **save**

1. save() возвращает сгенерированный идентификатор (ID) созданной сущности. persist() также сохраняет сущность в базе данных, но не возвращает ее ID.
2. Если переданная сущность уже имеет идентификатор, save() генерирует исключение. Если переданная сущность уже имеет идентификатор, persist() игнорирует переданную сущность и не выполняет запись в базу данных.
3. save() выполняет немедленно запись в базу данных. persist() может отложить операцию записи в базу данных до момента синхронизации сеанса Hibernate.

### Logging (log4j)



Популярные библиотеки логирования:

1. log4j
2. jdk14l
3. logback
4. simple

slf4j api – спецификация логирования, через этот интерфейс наша программа общается с разными библиотеками для логов.

Уровни логирования:

1. FATAL
2. ERROR
3. WARNING
4. WARN
5. INFO
6. DEGUB
7. TRACE (Самый детальный)