**Key Session methods in Hibernate 6.4.4**

Here’s a curated summary of the most-used methods on the Session object:

**1. Core CRUD operations**

* persist(Object entity) – JPA-standard: makes an entity persistent; insert may be delayed until flush/commit.
* save(Object entity) – Hibernate-specific: inserts immediately and returns the generated ID.
* update(Object entity) – Reattaches a detached entity; throws an exception if it's not in the database.
* merge(Object entity) – Merges changes from a detached object into the current session, returning a managed instance.
* remove(Object entity) – Deletes the specified entity row.
* get(Class<T>, id) – Loads eagerly; returns null if not found.
* getReference(Class<T>, id) (replacement for load) – Lazily fetches a proxy; fails if not found.

**2. Context & flushing**

* flush() – Synchronizes session changes to the database explicitly [Stack Overflow+12docs.jboss.org+12Wikipedia+12](https://docs.jboss.org/hibernate/orm/6.4/javadocs/org/hibernate/Session.html?utm_source=chatgpt.com)[Hibernate+4Hibernate+4Stack Overflow+4](https://hibernate.org/orm/documentation/6.4/?utm_source=chatgpt.com).
* setFlushMode(FlushModeType) / setHibernateFlushMode(FlushMode) / getFlushMode() – Controls when flush occurs (e.g. AUTO, COMMIT, MANUAL) .

**3. Read‑only & dirty checking**

* setDefaultReadOnly(boolean) – Determines whether loaded entities are read‑only by default.
* setReadOnly(Object entity, boolean) / isReadOnly(Object entity) – Manage read‑only state per entity, skipping dirty checking [docs.jboss.org](https://docs.jboss.org/hibernate/orm/6.4/javadocs/org/hibernate/Session.html?utm_source=chatgpt.com).
* isDirty() – Checks if the session has unsynchronized changes [Hibernate+2docs.jboss.org+2docs.jboss.org+2](https://docs.jboss.org/hibernate/orm/6.4/javadocs/org/hibernate/Session.html?utm_source=chatgpt.com).

**4. Fetch profiles**

* enableFetchProfile(String) / disableFetchProfile(String) / isFetchProfileEnabled(String) – Activate preconfigured fetch profiles for entity loading [docs.jboss.org](https://docs.jboss.org/hibernate/orm/6.4/javadocs/org/hibernate/Session.html?utm_source=chatgpt.com).

**5. Session management**

* clear() – Clears the persistence context (evicts all managed entities).
* evict(Object entity) – Detaches a specific entity from the session.
* close() – Ends the session and releases JDBC resources (also inherited via AutoCloseable).

**6. Utility & advanced**

* cancelQuery() – Cancels a running query (safe to call from another thread) [docs.jboss.org+3docs.jboss.org+3docs.jboss.org+3](https://docs.jboss.org/hibernate/orm/6.4/javadocs/org/hibernate/Session.html?utm_source=chatgpt.com).
* getStatistics() – Returns session-specific metrics.
* byId(Class<T>) – Fluent loader API.
* doWork(Work work) – Low-level JDBC access using the session’s connection [docs.jboss.org](https://docs.jboss.org/hibernate/orm/6.4/javadocs/org/hibernate/Session.html?utm_source=chatgpt.com).
* addEventListeners(...) – Register session-level event hooks.
* Locking support: buildLockRequest(LockOptions), etc.

**🧭 Tip for developers**

* Prefer **JPA-aligned methods** (persist, remove, getReference) for portable code.
* Use **Hibernate-specific methods** (save, update, cancelQuery, etc.) when you need advanced behavior.
* Control flushing and read‑only mode to optimize performance (especially for bulk processing).
* Manage session lifespan tightly—sessions are **not thread-safe** and should be short-lived (per transaction) [docs.jboss.org](https://docs.jboss.org/hibernate/orm/6.4/javadocs/org/hibernate/Session.html?utm_source=chatgpt.com).