Springboot: First Level Caching

JPA - PART3 (First-Level Cache)

application.properties

```
#database connection
spring.datasource.url=jdbc:h2:mem:userDB
spring.datasource.driver-class-name=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=


# Enable the H2 console and set the path
spring.h2.console.enabled=true
spring.h2.console.path=/h2-console

spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
```

```
@Entity
public class UserDetails {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String name;
    private String email;

    // Constructors
    public UserDetails() {
    }

    public UserDetails(String name, String email) {
        this.name = name;
        this.email = email;
    }

    // Getters and setters
}
```

Controller class

```
@RestController
@RequestMapping(value = "/usi/")
public class UserController {

    @Autowired
    UserDetailsService userDetailsService;

    @GetMapping(path = "/test-jpa")
    public UserDetails getUser() {
        UserDetails userDetails = new UserDetails( name: "xyx",
                email: "xyz@conceptandcoding.com");
        userDetailsService.saveUser(userDetails);
        UserDetails output1 =  userDetailsService.getUser( primaryKey: 1L);
        return output1;
    }

    @GetMapping(path = "/read-jpa")
    public UserDetails getUser2() {
        UserDetails output1 =  userDetailsService.getUser( primaryKey: 1L);
        return output1;
    }
}
```

```
@Repository
public interface UserDetailsRepository extends
        JpaRepository<UserDetails, Long> {
}
```

- Internally using EntityManager API only, but it provides some predefined methods like findAll, deleteAll etc.
- Also provides pagination and sorting capability...
- Also @Transactional support on operations which needs it on operation like Insert, modify, delete
- Start and close EntityManager resource, so no need to manage it.

```
@Service
public class UserDetailsService {

    @Autowired
    UserDetailsRepository userDetailsRepository;

    public void saveUser(UserDetails user) {
        userDetailsRepository.save(user);
    }

    public UserDetails getUser(Long primaryKey) {
        return userDetailsRepository.findById(primaryKey).get();
    }
}
```

Creating EntityManager

SimpleJpaRepository.java

```
@Transactional
public <S extends T> S save(S entity) {
    Assert.notNull(entity,  message: "Entity must not be null");
    if (this.entityInformation.isNew(entity)) {
        this.entityManager.persist(entity);
        return entity;
    } else {
        return this.entityManager.merge(entity);
    }
}
```

'SessionImpl' is the Hibernate implementation for EntityManager APIs, creates PersistenceContext f or each EntityManager and is doing first-level caching in the map

StatefulPersistenceContext.java

```
@Override
public void addEntity(EntityKey key, Object entity) {
    EntityHolderImpl holder = EntityHolderImpl.forEntity(key, key.getPersister(), entity);
    final EntityHolderImpl oldHolder = getOrInitializeEntitiesByKey().putIfAbsent(
            key,
            holder
    );
    if (oldHolder != null) {
        assert oldHolder.entity == null || oldHolder.entity == entity;
        oldHolder.entity = entity;
        holder = oldHolder;
    }
    holder.state = EntityHolderState.INITIALIZED;
    final BatchFetchQueue fetchQueue = this.batchFetchQueue;
    if (fetchQueue != null) {
        fetchQueue.removeBatchLoadableEntityKey(key);
    }
}
```

> key = {EntityKey@10767} "EntityKey{com.conceptandcoding.learningspringboot.jpa.entity.UserDetails#1}"
> entity = {UserDetails@10768}
> id = {Long@14243} 1
> name = "xyx"
> email = "xyz@conceptandcoding.com"

StatefulPersistenceContext.java

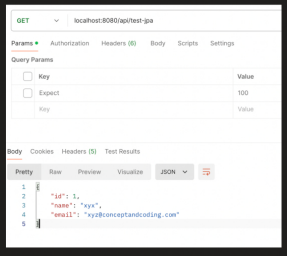
```
@Override
public EntityHolder claimEntityHolderIfPossible(
        EntityKey key,
        Object entity,
        JdbcValuesSourceProcessingState processingState,
        EntityInitializer initializer) {
    EntityHolderImpl holder = EntityHolderImpl.forEntity(key, key.getPersister(), entity);
    final EntityHolderImpl oldHolder = getOrInitializeEntitiesByKey().putIfAbsent(
            key,
            holder
    );

    if (oldHolder != null) {
        if (entity != null) {
            assert oldHolder.entity == null || oldHolder.entity == entity;
            oldHolder.entity = entity;
        }
        // Only setting a new entity initializer if there already is one owner
        // Else skip if an entity exists which is different from the effective optional object.
        // The effective optional object is the current object to be refreshed,
        // which always needs re-initialization, even if already initialized
        if (oldHolder.entityInitializer != null
                && oldHolder.entity != null && oldHolder.state != EntityHolderState.ZOMBIED_PROXY &&
                processingState.getProcessingOptions().getEffectiveOptionalObject() != null
                        && oldHolder.entity != processingState.getProcessingOptions().getEffectiveOptionalObject()
        ) {
            return oldHolder;
        }
        holder = oldHolder;
    }
    assert holder.entityInitializer == null || holder.entityInitializer == initializer;
    holder.entityInitializer = initializer;
    processingState.registerLoadingEntityHolder(holder);
    return holder;
}
```

> key = {EntityKey@10766} "EntityKey{com.conceptandcoding.learningspringboot.jpa.entity.UserDetails#1}"
> entity = null

During application startup (JPA creates fresh DB and table):

```
2024-11-11T15:00:46.273+05:30  INFO 9537 --- [           main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000489: No JTA platform available (set 'hibernate.transaction.jta.platform' t
Hibernate:
    drop table if exists user_details cascade
Hibernate:
    create table user_details (
        id bigint generated by default as identity,
        email varchar(255),
        name varchar(255),
        primary key (id)
    )
2024-11-11T15:00:46.285+05:30  INFO 9537 --- [           main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2024-11-11T15:00:46.374+05:30  WARN 9537 --- [           main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may b
2024-11-11T15:00:46.555+05:30  INFO 9537 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8080 (http) with context path '/'
2024-11-11T15:00:46.560+05:30  INFO 9537 --- [           main] c.c.l.SpringbootApplication              : Started SpringbootApplication in 1.542 seconds (process running for 1.7)
```

When invoked /test-jpa API



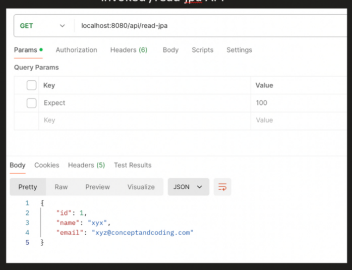

```
Hibernate:
    drop table if exists user_details cascade
Hibernate:
    create table user_details (
        id bigint generated by default as identity,
```

Report Abuse

```
        email varchar(255),
        name varchar(255),
        primary key (id)
    )
2024-11-11T15:02:28.409+05:30  INFO 9564 --- [          main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit '
2024-11-11T15:02:28.490+05:30  WARN 9564 --- [          main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, d
2024-11-11T15:02:28.647+05:30  INFO 9564 --- [          main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8080 (http) with context path '/'
2024-11-11T15:02:28.653+05:30  INFO 9564 --- [          main] c.c.l.SpringbootApplication              : Started SpringbootApplication in 1.484 seconds (process run
2024-11-11T15:02:32.097+05:30  INFO 9564 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring DispatcherServlet 'dispatcherServlet'
2024-11-11T15:02:32.097+05:30  INFO 9564 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
2024-11-11T15:02:32.098+05:30  INFO 9564 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Completed initialization in 1 ms
Hibernate:
    insert
    into
        user_details
        (email, name, id)
    values
        (?, ?, default)
```

Only insert call, even though we are doing first insert and then read

invoked /read-jpa API




```
2024-11-11T15:02:28.409+05:30  INFO 9564 --- [          main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit '
2024-11-11T15:02:28.490+05:30  WARN 9564 --- [          main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, d
2024-11-11T15:02:28.647+05:30  INFO 9564 --- [          main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8080 (http) with context path '/'
2024-11-11T15:02:28.653+05:30  INFO 9564 --- [          main] c.c.l.SpringbootApplication              : Started SpringbootApplication in 1.484 seconds (process run
2024-11-11T15:02:32.097+05:30  INFO 9564 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring DispatcherServlet 'dispatcherServlet'
2024-11-11T15:02:32.097+05:30  INFO 9564 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
2024-11-11T15:02:32.098+05:30  INFO 9564 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Completed initialization in 1 ms
Hibernate:
    insert
    into
        user_details
        (email, name, id)
    values
        (?, ?, default)
Hibernate:
    select
        ud1_0.id,
        ud1_0.email,
        ud1_0.name
    from
        user_details ud1_0
    where
        ud1_0.id=?
```

So, PersistenceContext scope is associated with EntityManager:

```java
@Service
public class UserDetailsService {


    @Autowired
    EntityManagerFactory entityManagerFactory;

    public UserDetails saveUser(UserDetails user) {

        EntityManager entityManager = entityManagerFactory.createEntityManager(); //session1 created
        entityManager.getTransaction().begin(); //transaction created
        entityManager.persist(user);
        entityManager.find(UserDetails.class, primaryKey: 1L);
        UserDetails output =  entityManager.find(UserDetails.class, primaryKey: 1L);
        System.out.println("i am able to find the data, name is:" + output.getName());
        entityManager.getTransaction().commit(); //transaction committed
        entityManager.close(); // session1 closed


        EntityManager entityManager2 = entityManagerFactory.createEntityManager(); //session2 created
        entityManager2.getTransaction().begin(); //transaction created
        entityManager2.find(UserDetails.class, primaryKey: 1L);
        UserDetails output2 =  entityManager2.find(UserDetails.class, primaryKey: 1L);
        System.out.println("Session2: i am able to find the data, name is:" + output.getName());
        entityManager2.getTransaction().commit(); //transaction committed
        entityManager2.close(); // session1 closed

        return output2;
    }
}
```

Output:

```
2024-11-11T16:58:58.995+05:30  INFO 10877 --- [          main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000489: No JTA platform available
Hibernate:
    drop table if exists user_details cascade
Hibernate:
    create table user_details (
        id bigint generated by default as identity,
        email varchar(255),
        name varchar(255),
        primary key (id)
    )
2024-11-11T16:58:59.006+05:30  INFO 10877 --- [          main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory
2024-11-11T16:58:59.025+05:30  WARN 10877 --- [          main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by
2024-11-11T16:58:59.216+05:30  INFO 10877 --- [          main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8080 (http) wi
2024-11-11T16:58:59.241+05:30  INFO 10877 --- [          main] c.c.l.SpringbootApplication              : Started SpringbootApplication in 1.39
2024-11-11T16:59:01.873+05:30  INFO 10877 --- [nio-8080-exec-2] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring DispatcherServlet
2024-11-11T16:59:01.873+05:30  INFO 10877 --- [nio-8080-exec-2] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServl
2024-11-11T16:59:01.873+05:30  INFO 10877 --- [nio-8080-exec-2] o.s.web.servlet.DispatcherServlet        : Completed initialization in 0 ms
Hibernate:
    insert
    into
        user_details
        (email, name, id)
    values
        (?, ?, default)
```

```
i am able to find the data, name is:xyx
Hibernate:
    select
        ud1_0.id,
        ud1_0.email,
        ud1_0.name
    from
        user_details ud1_0
    where
        ud1_0.id=?
Session2: i am able to find the data, name is:xyx
```

And EntityManager is created for each HTTP Request, so different methods within same HTTP Request share the EntityManager

DispatcherServelt Code snapshot:

```
if (!mappedHandler.applyPreHandle(processedRequest, response)) {
    return;
}

// Actually invoke the handler.
mv = ha.handle(processedRequest, response, mappedHandler.getHandler());

if (asyncManager.isConcurrentHandlingStarted()) {
    return;
}
```

```
@Override
public void preHandle(WebRequest request) throws DataAccessException {
    String key = getParticipateAttributeName();
    WebAsyncManager asyncManager = WebAsyncUtils.getAsyncManager(request);
    if (asyncManager.hasConcurrentResult() && applyEntityManagerBindingInterceptor(asyncManager, key)) {
        return;
    }

    EntityManagerFactory emf = obtainEntityManagerFactory();
    if (TransactionSynchronizationManager.hasResource(emf)) {
        // Do not modify the EntityManager: just mark the request accordingly.
        Integer count = (Integer) request.getAttribute(key, WebRequest.SCOPE_REQUEST);
        int newCount = (count != null ? count + 1 : 1);
        request.setAttribute(getParticipateAttributeName(), newCount, WebRequest.SCOPE_REQUEST);
    }
    else {
        logger.debug("Opening JPA EntityManager in OpenEntityManagerInViewInterceptor");
        try {
            EntityManager em = createEntityManager();
            EntityManagerHolder emHolder = new EntityManagerHolder(em);
            TransactionSynchronizationManager.bindResource(emf, emHolder);

            AsyncRequestInterceptor interceptor = new AsyncRequestInterceptor(emf, emHolder);
            asyncManager.registerCallableInterceptor(key, interceptor);
            asyncManager.registerDeferredResultInterceptor(key, interceptor);
        }
        catch (PersistenceException ex) {
            throw new DataAccessResourceFailureException("Could not create JPA EntityManager", ex);
        }
    }
}
```

```
@RestController
@RequestMapping(value = "/api/")
public class UserController {

    @Autowired
    UserDetailsService userDetailsService;

    @GetMapping(path = "/test-jpa")
    public UserDetails getUser() {
        UserDetails userDetails1 = new UserDetails( name: "xyx",
                email: "xyz@conceptandcoding.com");

        userDetailsService.saveUser(userDetails1);
        return userDetailsService.findById( primaryKey: 1L);
    }

}
```

```
@Service
public class UserDetailsService {

    @Autowired
    UserDetailsRepository userDetailsRepository;

    public void saveUser(UserDetails user) {

        userDetailsRepository.save(user);
    }

    public UserDetails findById(Long primaryKey) {
        return  userDetailsRepository.findById(primaryKey).get();
    }
}
```

Internally both uses the same Entity Manager, therefore when findById invoked, Cache Hit will happen

Output:

```
2024-11-11T18:35:19.899+05:30  INFO 11152 --- [          main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH0
Hibernate:
    drop table if exists user_details cascade
Hibernate:
    create table user_details (
        id bigint generated by default as identity,
        email varchar(255),
        name varchar(255),
        primary key (id)
    )
2024-11-11T18:35:19.106+05:30  INFO 11152 --- [          main] j.LocalContainerEntityManagerFactoryBean : Init
2024-11-11T18:35:19.106+05:30  WARN 11152 --- [          main] JpaBaseConfiguration$JpaWebConfiguration : spri
2024-11-11T18:35:19.343+05:30  INFO 11152 --- [          main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tom
2024-11-11T18:35:19.347+05:30  INFO 11152 --- [          main] c.c.1.SpringBootApplication              : Star
2024-11-11T18:35:24.722+05:30  INFO 11152 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Init
2024-11-11T18:35:24.722+05:30  INFO 11152 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Init
2024-11-11T18:35:24.722+05:30  INFO 11152 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Comp
Hibernate:
    insert
    into
        user_details
        (email, name, id)
    values
        (?, ?, default)
```