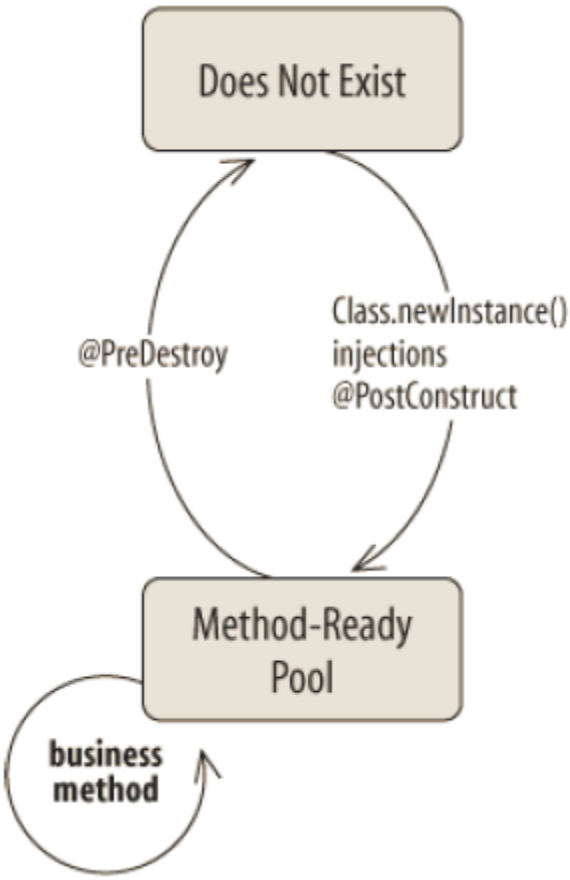
Enterprise Java Beans

EJB Typen

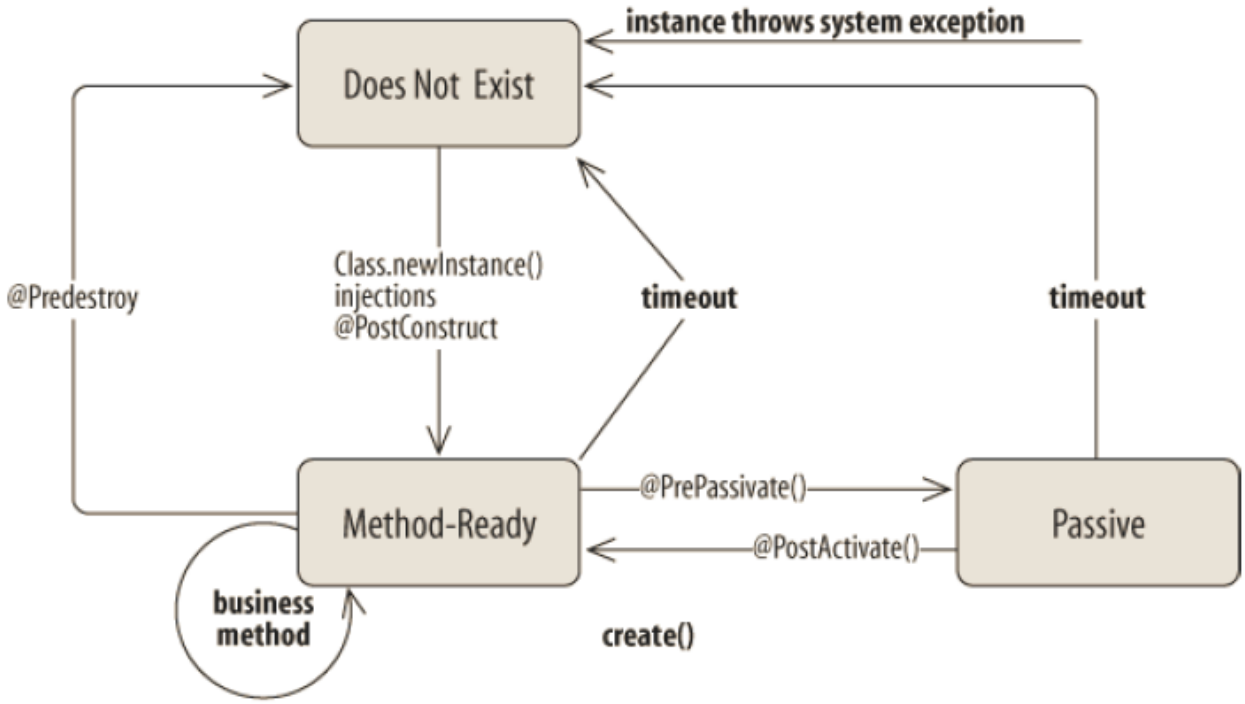
- Session Beans
  - Stateless
  - Stateful
  - Singletonn
- Message Driven Beans
- Timer Service

Lifecyle

Stateless



Stateful



Singletons

Concurrency

```
@Singleton
public class DiscountRateBean implements DiscountRateLocal {
    ...
    @Lock(READ)
    public Rate getRate() { return rate; }
    @Lock(WRITE)
    public void setRate(Rate rate) { this.rate = rate; }
}
```

Asynchronous

@

Timer

Programming a Timer

```
@Stateless
public class ShipMaintenanceBean implements ShipMaintenanceLocal {
   @Resource TimerService timerService;
   public void scheduleMaintenance(Ship ship, ScheduleExpression sched) {
      timerService.createCalendarTimer(sched, new TimerConfig(ship, true));
   }
   @Timeout
   public void maintenance(Timer timer) {
      Ship ship = (Ship)timer.getInfo();
      ...
   }
}
```

Schedule

```
@Stateless
public class ShipMaintenanceBean implements ShipMaintenanceLocal {
   @Schedule(year="*", month="*", dayOfMonth="20")
   public void maintenance(Timer timer) {
      ...
   }
}
```

Interceptors

Interceptor Class

```
public class Profiler {
   @AroundInvoke
   public Object profile(InvocationContext invocation) throws Exception {
      long startTime = System.currentTimeMillis();
      try { return invocation.proceed(); }
      finally {
         long time = System.currentTimeMillis() - startTime;
         System.out.println(invocation.getMethod() + ": " + time + "ms");
      }
   }
}
```

Applying Interceptors

```
@Stateful
@Interceptors(Logger.class)
public class TravelAgentBean implements TravelAgentRemote {
   ...
   @Interceptors(Profiler.class)
   public TicketDO bookPassage(CreditCardDO card, double price)
      throws IncompleteConversationalState {
      ...
   }
}
```

Default Interceptors

```
<ejb-jar version="3.1" xmlns="http://java.sun.com/xml/ns/javaee" ...>
   <interceptors>
      <interceptor>
         <interceptor-class>com.titan.Profiler</interceptor-class>
      </interceptor>
   </interceptors>
   <assembly-descriptor>
      <interceptor-binding>
         <ejb-name>*</ejb-name>
         <interceptor-class>com.titan.Logger</interceptor-class>
      </interceptor-binding>
   </assembly-descriptor>
</ejb-jar>
```

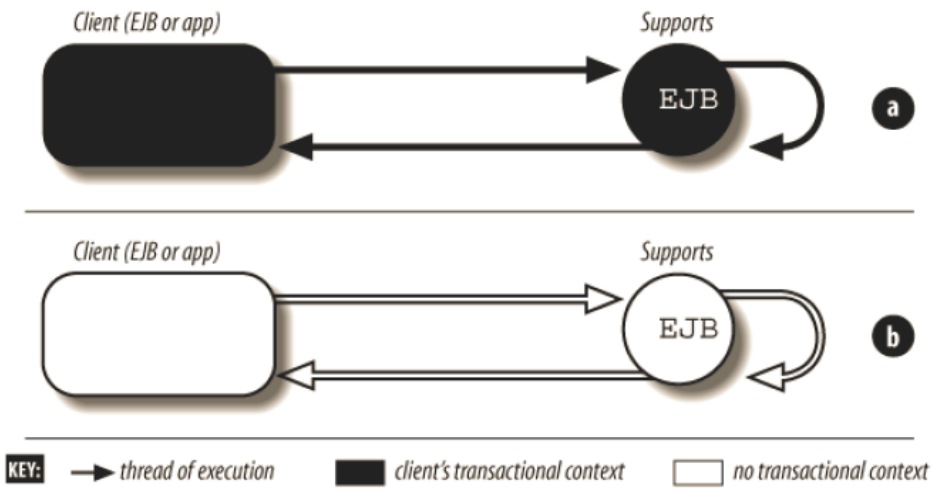
Transactions

```
@Stateful
@TransactionAttribute(NOT_SUPPORTED)
public class TravelAgentBean implements TravelAgentRemote {
   public Customer findCustomer(String firstName, String lastName) { ... }
   public void updateAddress(Address address) { ... }
   ...
   @TransactionAttribute(REQUIRED)
   public TicketDO bookPassage(CreditCardDO card, double price) { ... }
}
```

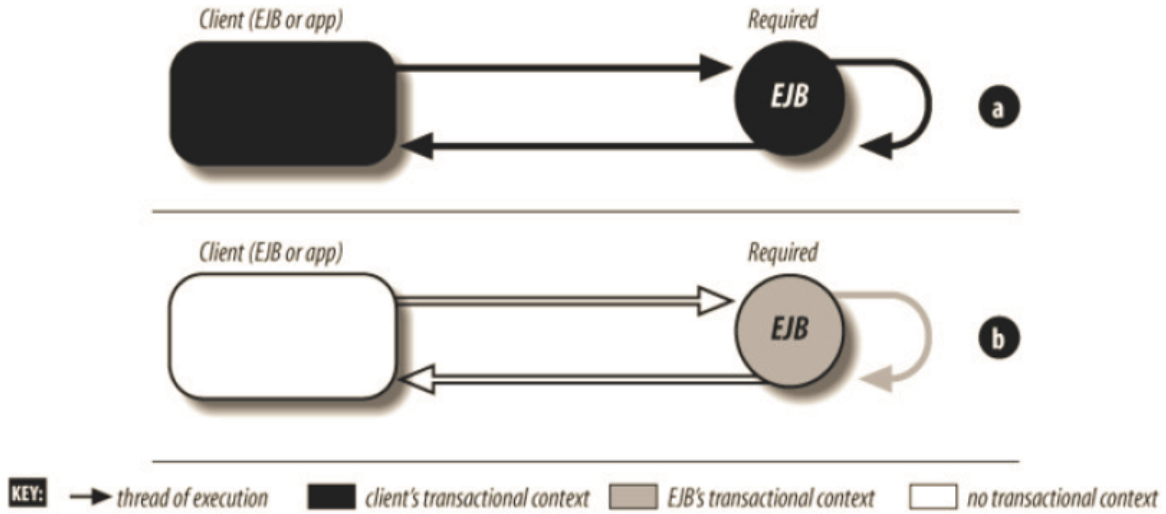
Transaction Attributes

NotSupported

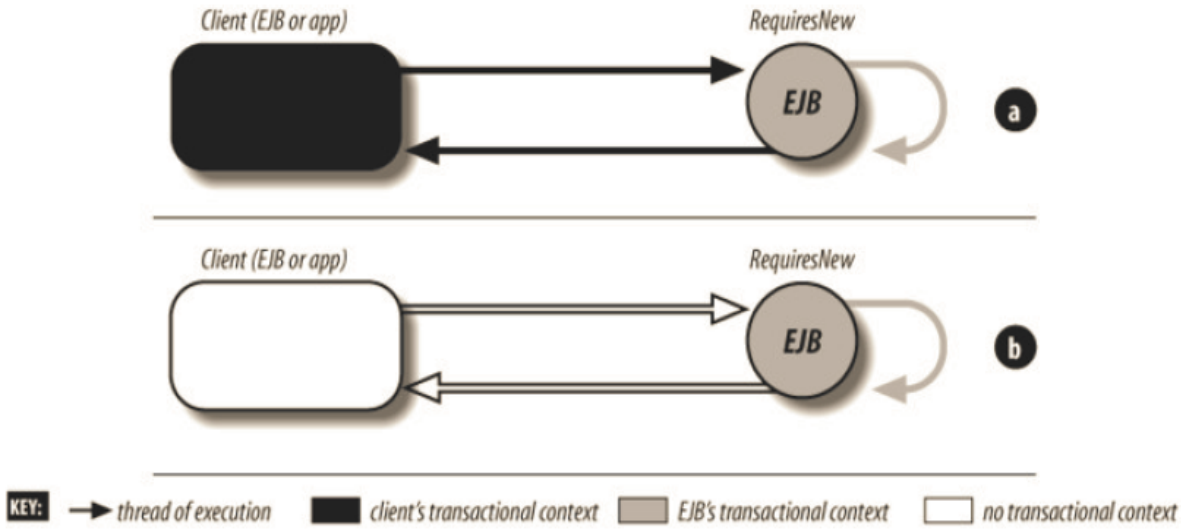
Supports



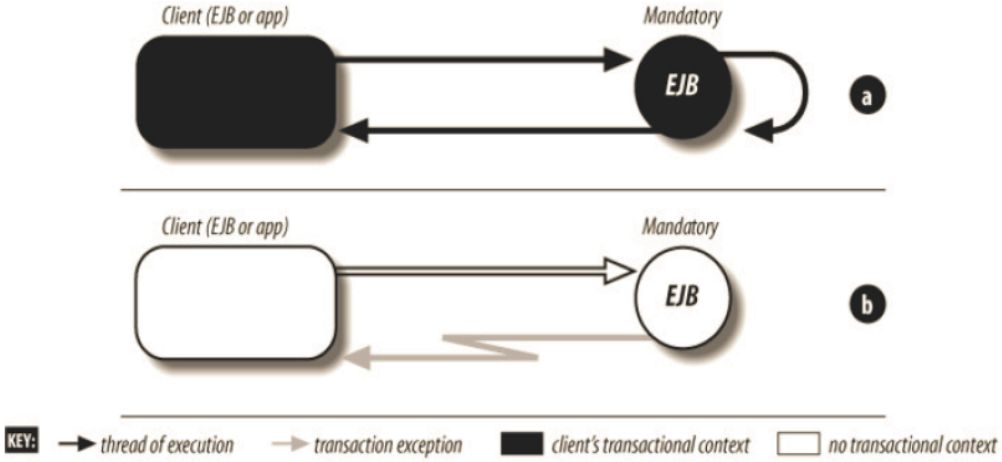
Required



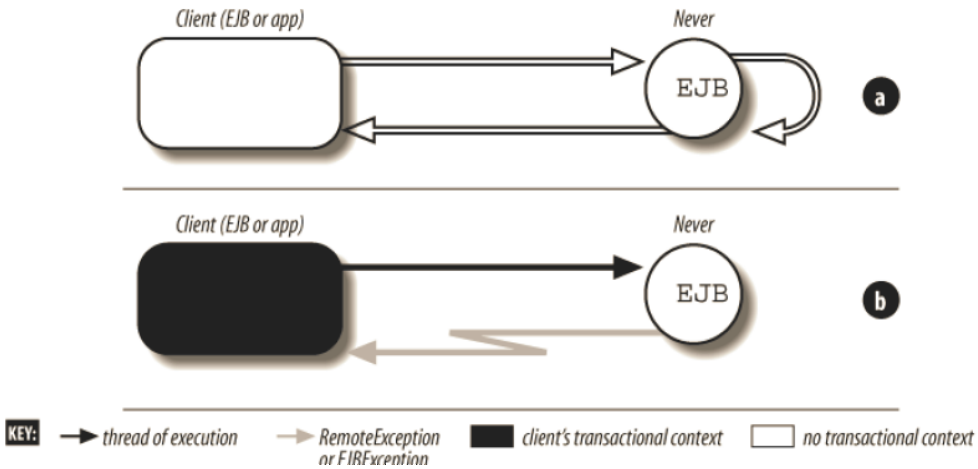
Requries New



Mandatory



Never



Bean Managed Transactions

```
@Stateful
@TransactionManagement(BEAN)
public class TravelAgentBean implements TravelAgentRemote {
   @Resource UserTransaction transaction;
   public TicketDO bookPassage(CreditCardDO card, double price) {
      transaction.begin();
      ...
      transaction.commit();
   }
}
```

Security

Permissions

```
@Stateless
@RolesAllowed("AUTHORIZED_AGENT")
public class ProcessPaymentBean implements ProcessPayment {
    ...
    public boolean byCredit(...) throws PaymentException { ... }
    @RolesAllowed("CHECK_FRAUD")
    public boolean byCheck(...) throws PaymentException { ... }
    @PermitAll
    public boolean byCash(...) throws PaymentException { ... }
}
```

Programmatic Security

```
@Stateless
@DeclareRoles("JUNIOR_AGENT")
public class ProcessPaymentBean implements ProcessPaymentLocal {
    @Resource
    private SessionContext sessionContext;
    ...
    public void byCredit(Customer customer, CreditCardDO card,
        double amount) throws PaymentException {
        if (sessionContext.isCallerInRole("JUNIOR_AGENT") &&
            amount > maxJuniorTrade) throw new PaymentException(...);
        String agent = sessionContext.getCallerPrincipal().getName();
        ...
    }
}
```