**Sequence Diagram Notations**

**Lifeline**

A lifeline represents an individual participant in the interaction.



**Actor**

An actor is a role played by an entity interacting with the subject. It can represent human users, external hardware, or other subjects.

**Activation**

An activation, represented by a thin rectangle on a lifeline, signifies the period during which an element is performing an operation.

UML Sequence Diagram: Actor example

**Messages**

**Call Message**

A call message defines communication between lifelines, representing the invocation of an operation on the target lifeline.

UML Sequence Diagram: Call message example

**Return Message**

A return message represents the passing of information back to the caller of a corresponding former message.

UML Sequence Diagram: Return message example

**Self Message**

A self message denotes communication within the same lifeline, representing the invocation of a message on itself.

UML Sequence Diagram: Self message example

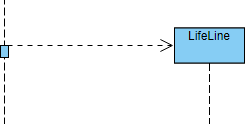
**Recursive Message**

A recursive message is similar to a self message but points to an activation on top of the current one.

UML Sequence Diagram: Recursive message example

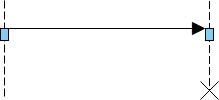
**Create Message**

A create message signifies the instantiation of a target lifeline.



**Destroy Message**

A destroy message represents the request to destroy the lifecycle of the target lifeline.



**Duration Message**

A duration message shows the time distance between two time instants for a message invocation.



**Note**

A note or comment provides the ability to attach remarks to elements but carries no semantic force.

UML Sequence Diagram: Note example

**Drawing a Sequence Diagram: Step-by-Step Guide**

1. **Identify Participants:**
   * Determine the objects participating in the collaboration or use case scenario.
2. **Scenario Analysis:**
   * If derived from a scenario, select the normal scenarios first.
   * Identify the primary actor(s) activating the use case.
3. **Message Flow:**
   * Consider the initiating point of the scenario.
   * Define the system’s response to the actor’s message and what needs handling before the return message.
4. **Object and Operation Identification:**
   * Identify candidate objects and operations based on the scenario.
   * Use this information to incrementally derive the class diagram.
5. **Repeat Scenario Points:**
   * Iterate through each point of the scenario until completion.
6. **Alternative Scenarios:**
   * Draw corresponding sequence diagrams for exception or alternative scenarios.