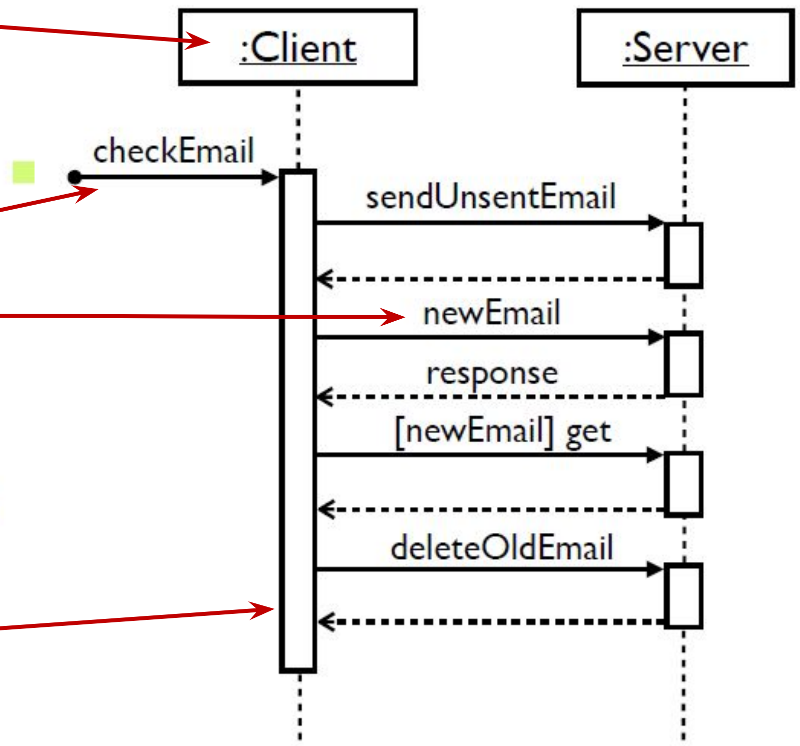
Sequence diagrams

- **Sequence diagram**: an “interaction diagram” that models a single scenario executing in a system
- Shows what messages are sent and when
- Visualizes the execution sequences of an use case

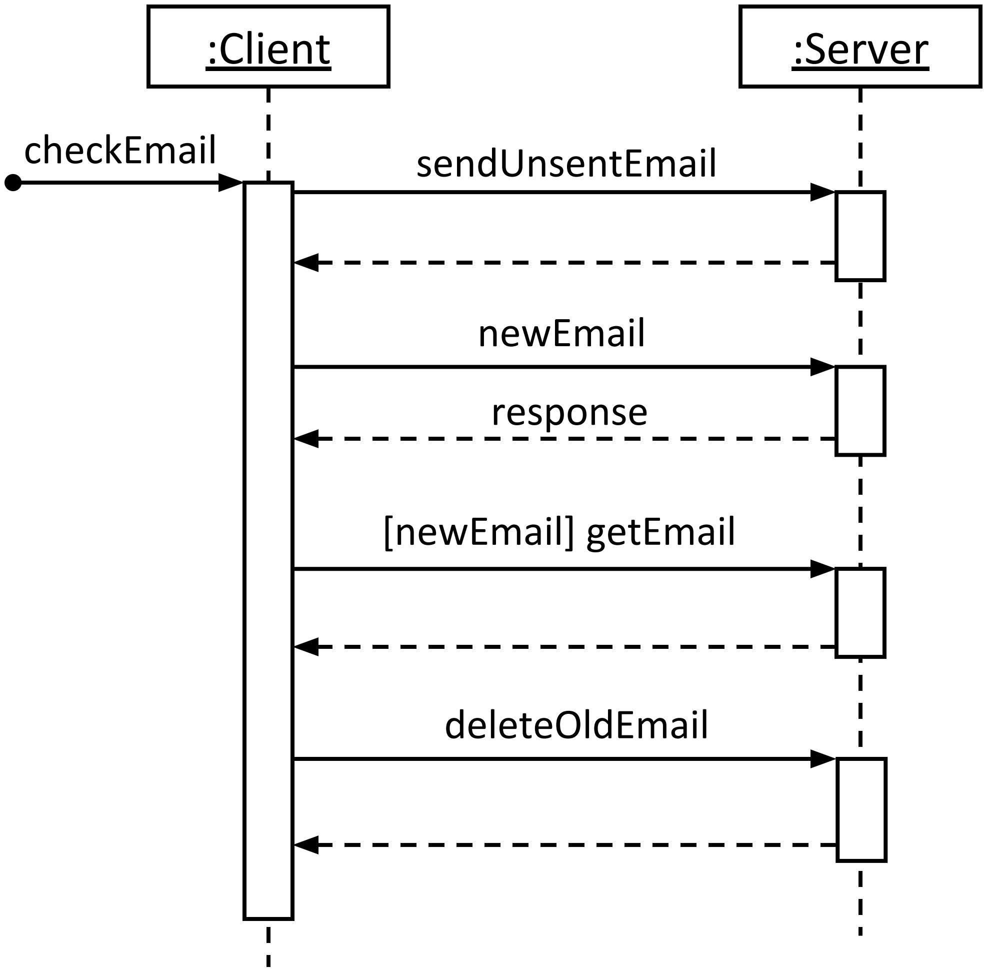
Key parts of a sequence diagram

- **Participant:** an object or an entity; the sequence diagram actor
  - sequence diagram starts with an unattached "found message" arrow
- **Message:** communication between objects
- Axes in a sequence diagram:
  - **horizontal:** which participant is acting
  - **vertical:** time (↓ forward in time)



Sequence diagram from a use case

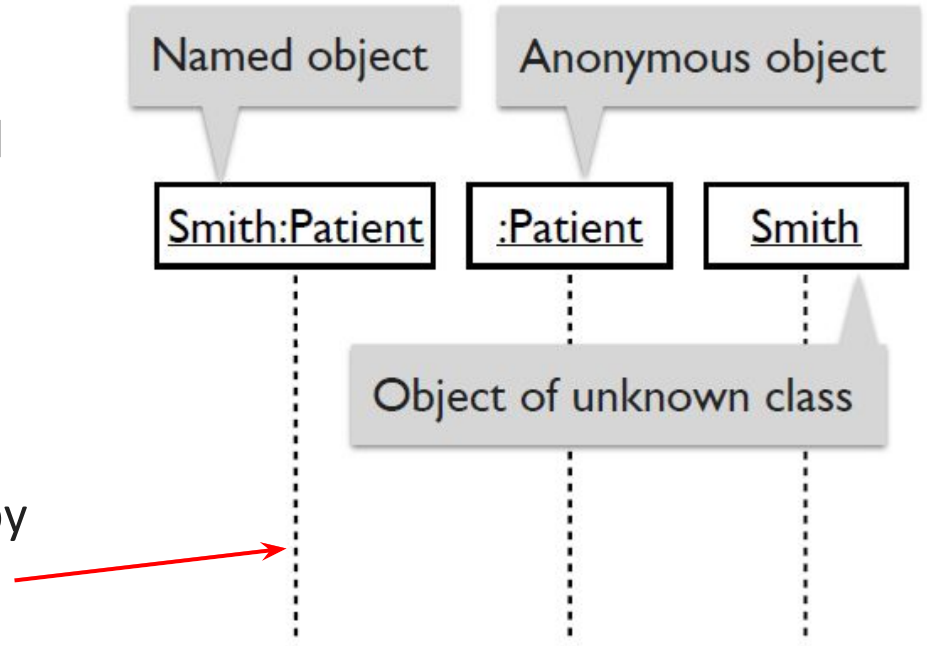
1. The user presses the “check email” button.
2. The client first sends all unsent email to the server.
3. After receiving an acknowledgement, the client asks the server if there is any new email.
4. If so, it downloads the new email.
5. Next, it deletes old thrashed email from the server.



Syntax: Representing objects

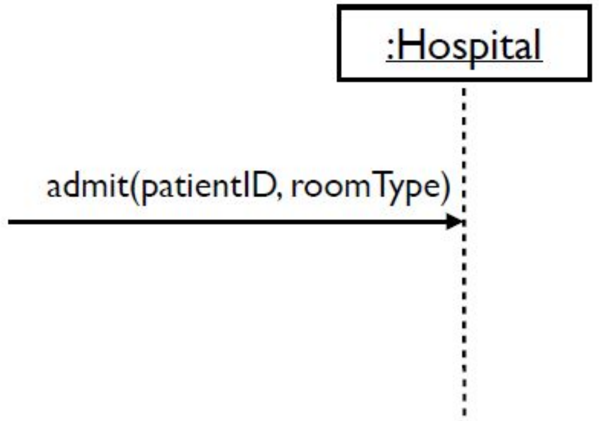
<u>**Objectname:classname**</u>

- An **object**: a **box** with an underlined label that specifies the object type, and optionally the object name.
  - Write the object's name if it clarifies the diagram.
- An object's "life line" is represented by a dashed vertical line.
  - Represents the life span of the object during the scenario being modeled

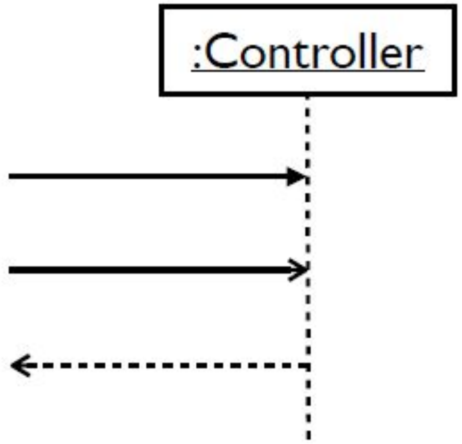


Representing messages between objects

- A **message** (method call): **horizontal arrow** to the receiving object.
  - Write message name and arguments above the arrow.

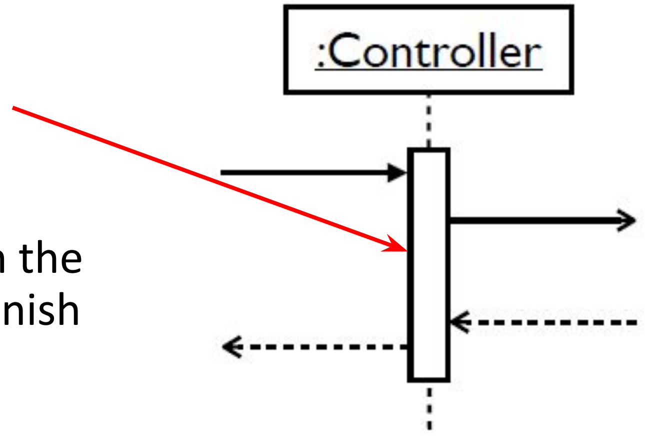


- Type of arrow indicates types of messages:
  - Synchronous message: solid arrow with a solid head.
  - Asynchronous message: solid arrow with a stick head.
  - Return message: dashed arrow with stick head.

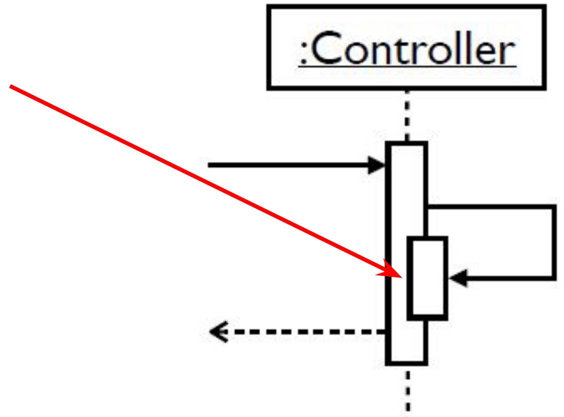


Indicating method execution

- **Activation**: thick box over object's life line, drawn when an object's method is on the stack
  - Either that object is running its code, or it is on the stack waiting for another object's method to finish

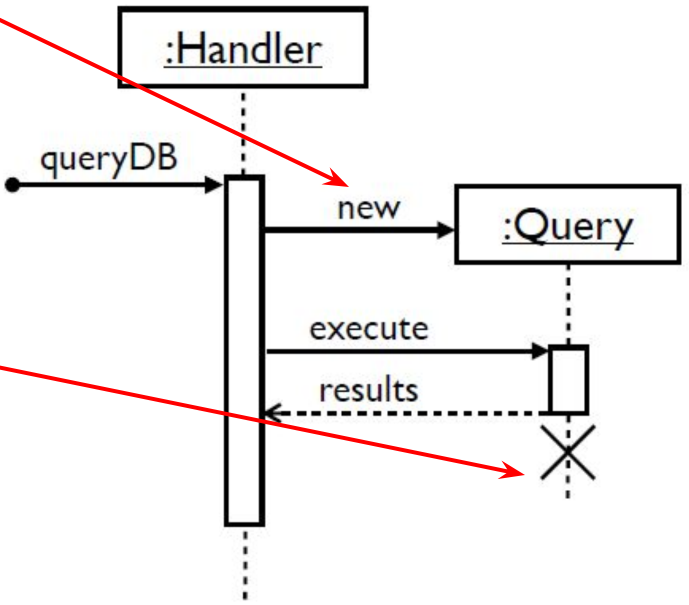


- Nest activations to indicate an object calling itself.



Lifetime of objects

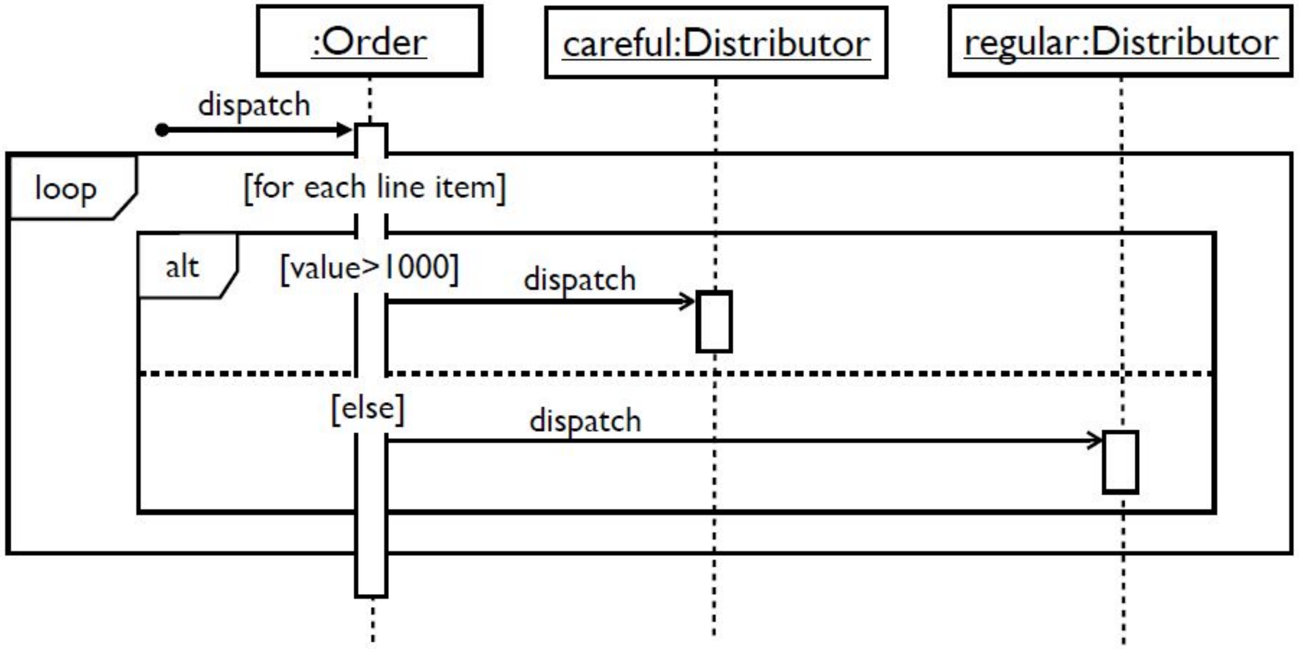
- Object **creation**: an arrow with **new** written above it
  - An object created after the start of the scenario appears lower than the others.
- Object **deletion**: **X** at the bottom of object's lifeline
  - Java doesn't explicitly delete objects; they fall out of scope and are garbage collected.



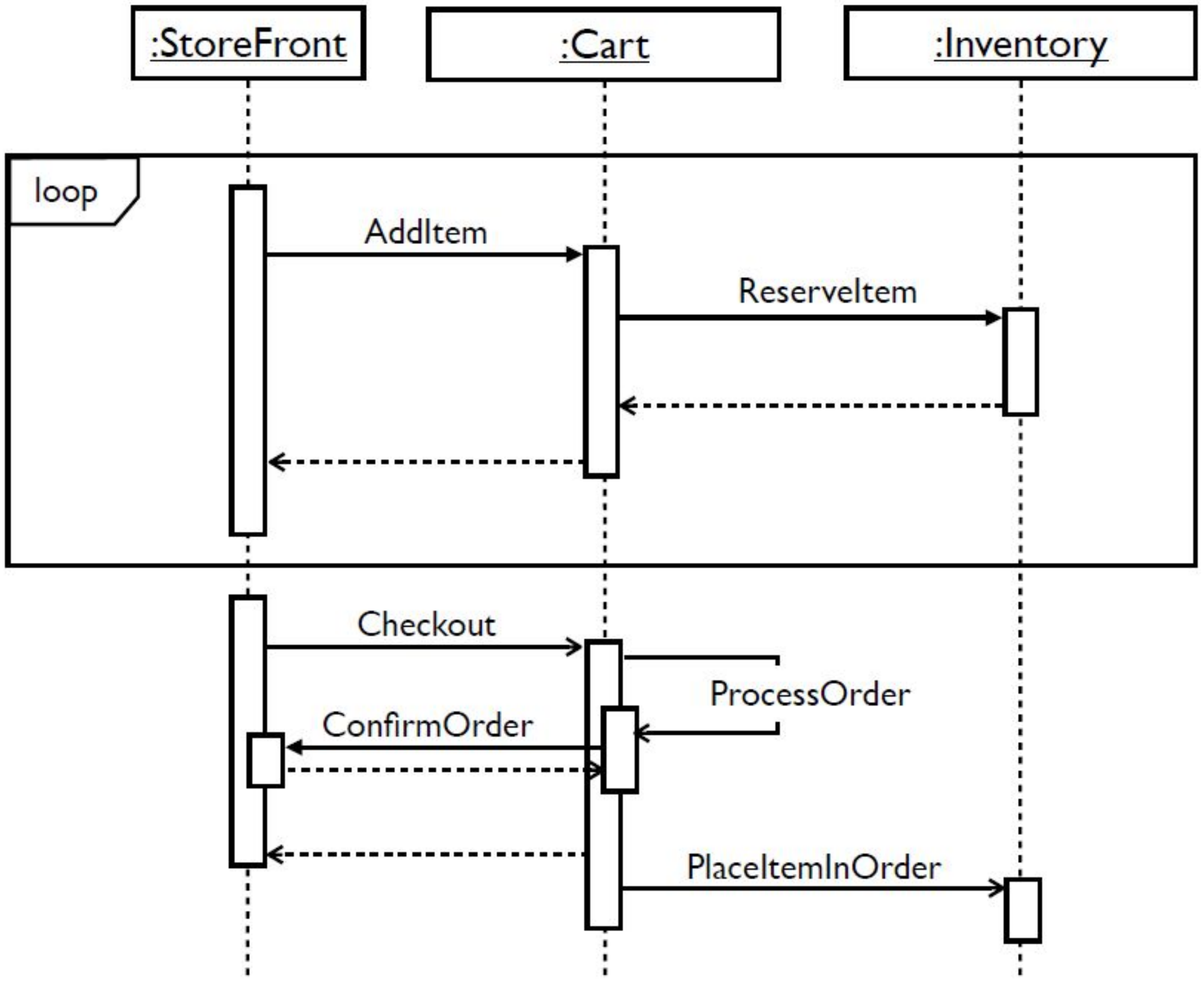
Alternatives, options, and loops

Frame: a box around part of a sequence diagram

- ❑ if → (opt) [condition]
- ❑ if/else→ (alt) [condition], separated by horizontal dashed line
- ❑ loop → (loop) [condition or items to loop over]



Example: Sequence diagram



Example

- https://creately.com/blog/diagrams/sequence-diagram-tutorial/#:~:text=get%20something%20done.-,Sequence%20Diagram%20Notations,them%20are%20represented%20by%20arrows.