Diagramma di Sequenza (Dynamic View & Message Passing)

Riferimento: `diagramma di sequenza.pdf`

Il diagramma modella il **flusso dei messaggi** nel tempo per lo scenario *Use Case: Create Order*, evidenziando la separazione tra i layer MVC.

- **Layer Frontend (Client-Side):**
  - **Lifeline `Cliente.html/js`:** L'attore innesca un evento asincrono ("Clicca Conferma Ordine"). Il browser non ricarica la pagina ma invia una richiesta **XHR/Fetch (POST)** con payload JSON `application/json` contenente `{ id_esercente, items: [...] }`.
- **Layer Controller (Server-Side Entry Point):**
  - **Lifeline `create.php`:** Agisce da *Controller*. Riceve il JSON, lo deserializza e istanzia la connessione al DB.
  - **Loop di Validazione (Security):** Prima di creare l'ordine, esegue un ciclo *self-call* o verso il DB per ricalcolare il totale server-side, prevenendo *Parameter Tampering* (modifica prezzi da parte del client).
- **Layer Domain (Business Logic):**
  - **Messaggio `create()`:** Il Controller delega alla Factory.
  - **Messaggio `process()`:** L'oggetto `TakeawayOrder` prende il controllo del flusso.
  - **Transazione Database:** Si nota l'apertura esplicita di una transazione (`BEGIN TRANSACTION`). Vengono eseguite sequenzialmente:
    1. `INSERT INTO ORDINE` (Testata).
    2. `INSERT INTO RIGA_ORDINE` (Iterazione sui prodotti).
    3. `COMMIT` (Solo se tutto ha successo).
- **Return Value:** Il sistema restituisce un oggetto strutturato `{ success: true, codice_ritiro: "XYZ" }` con status HTTP **201 Created**.