Simplified Digital Wallet

Problem Statement

You are tasked with designing a simplified digital wallet system. The wallet should allow users to store money (represented as a `double`), perform basic transactions with fees, and have limits on withdrawals.

Core Classes

Transaction Class

Attributes:

- `type (String)`: Represents the type of transaction, either `"DEPOSIT"` or `"WITHDRAW"`.
- `amount (double)`: Represents the amount of money involved in the transaction.
- `fee (double)`: Represents the fee applied to the transaction.

Methods:

- **Constructor**: Should take in the `type`, `amount`, and `fee`.

Wallet Class

Attributes:

- `balance (double)`: Represents the current amount of money in the wallet. It should be initialized to `0.0` by default.
- `transactions (List<Transaction>)`: A list to store all transaction objects done on this wallet.
- `withdrawalLimit (double)`: The maximum amount allowed for a single withdrawal transaction. This should be initialized by the constructor.
- `withdrawalFeePercentage (double)`: The percentage fee applied to each withdrawal. This should be initialized by the constructor.

Methods:

- **`deposit(amount: double) -> boolean`**
  - Adds the given `amount` to the wallet's balance.
  - If the `amount` is negative, return `false` without adding the amount and without creating a transaction.
  - If successful, create a new transaction with type `"DEPOSIT"` (fee should be `0`), add it to the transaction list, and return `true`.
- **`withdraw(amount: double) -> boolean`**
  - Subtracts the given `amount` from the wallet's balance.
  - If the `amount` is negative, greater than the current balance, or greater than `withdrawalLimit`, return `false` without subtracting any amount or creating a transaction.
  - If successful, create a new transaction with type `"WITHDRAW"`, calculate the fee as `withdrawalFeePercentage` of the `amount`, add it to the transaction list, and return `true`.
- **`getBalance() -> double`**
  - Returns the current balance of the wallet.
- **`getTransactions() -> List<Transaction>`**
  - Returns the list of all transactions done on this wallet.
- **Constructor**
  - The constructor should take in `withdrawalLimit` and `withdrawalFeePercentage`.

Requirements

- Ensure the **Wallet** class correctly handles deposits and withdrawals using **if-else** conditions.
- Proper validation for **invalid amounts**, **withdrawal limits**, and **insufficient balance**.
- The **Transaction** class should have correct attributes.
- The transactions list should be maintained correctly in the **Wallet**.
- **Withdrawal fee** should be a percentage of the withdrawal amount and should be calculated and added to the transaction object as a `fee`.
- The **withdrawal limit** and **fee percentage** should be set during the constructor.
- **No abstract classes, exceptions, or timestamps** should be used.