## LocalDateTime with formatter

```
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.time.format.DateTimeParseException;

public class Main {
    public static void main(String[] args) {
        String dateString = "2024-02-26";

        try {
            // Parse the string into a LocalDate object using the ISO_DATE format
            LocalDate date = LocalDate.parse(dateString);
            System.out.println("Parsed LocalDate: " + date);

            // If you want to specify a custom date format, you can use a
DateTimeFormatter
            DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-
MM-dd");
            LocalDate customDate = LocalDate.parse(dateString, formatter);
            System.out.println("Parsed Custom LocalDate: " + customDate);
        } catch (DateTimeParseException e) {
            // Handle parsing errors
            System.out.println("Error parsing date: " + e.getMessage());
        }
    }
}
```

- We have a string dateString containing a date in the format "YYYY-MM-DD" (year-month-day).
- We use LocalDate.parse() to parse this string into a LocalDate object. By default, it uses the ISO_DATE format ("YYYY-MM-DD").
- We catch a DateTimeParseException in case the input string is not in the expected format or cannot be parsed into a LocalDate.
- Optionally, we can use a DateTimeFormatter to specify a custom date format if the input string doesn't match the ISO_DATE format. We pass this formatter as a second argument to LocalDate.parse().