Processing Input Data

Michael Hoffman

DEVELOPER AND PLURALSIGHT AUTHOR

@mhi_inc   github.com/michaelhoffmantech

Item Processor

Apply business rules repeatedly

Transform input data

Validation

PatientEntity Class

```java
@Entity
@Table(name = "patient")
public class PatientEntity implements Serializable {

    @Id
    @GeneratedValue(strategy = GenerationType.SEQUENCE,
            generator = "sequenceGenerator")
    @SequenceGenerator(name = "sequenceGenerator")
    @Column(name = "patient_id")
    private Long id;

    @NotNull
    @Column(name = "source_id", nullable = false)
    private String sourceId;

    …

}
```

Demo

Demo 15 – Creating the entity type for transformation

```java
@Bean
@StepScope
public Function<PatientRecord, PatientEntity> processor() {
    return (patientRecord) ->  {
        return new PatientEntity(
            patientRecord.getSourceId(),
            patientRecord.getFirstName(),

            …

            LocalDate.parse(patientRecord.getBirthDate(),
                DateTimeFormatter.ofPattern("M/dd/yyyy"))
    };
}
```

BatchJobConfiguration.java

Package com.pluralsight.springbatch.patientbatchloader.config

Demo

Demo 16 – Implementing the item processor

```java
@Test
public void testProcessor() throws Exception {

        PatientRecord patientRecord = new PatientRecord( … );
        PatientEntity entity = processor.apply(patientRecord);
        assertNotNull(entity);
        assertEquals("72739d22-3c12-539b-b3c2-13d9d4224d40", entity.getSourceId());

        …

}
```

BatchJobConfigurationTest.java

Class in the test folder under the package
  com.pluralsight.springbatch.patientbatchloader.config

Demo

Demo 17 – Testing the item processor

Demo

Demo 18 – Executing the job with the item processor

Summary

Added a new entity type

Implemented an item processor

Tested the item processor