|  |  |
| --- | --- |
| problem statement:  1. Implement CsvConverter class  2. Implement StandardCsvConverter class  3. Implement JUnit 5 tests for CsvConverter in CsvConverterTests class  4. Implement JUnit 5 tests for StandardCsvConverter in StandardCsvConverterTests class  Notes:  1. New interfaces/classes can be added  2. New library dependencies can be added | постановка задачи:  1. Реализовать класс CsvConverter  2. Реализовать класс StandardCsvConverter  3. Реализуйте тесты JUnit 5 для CsvConverter в классе CsvConverterTests.  4. Реализуйте тесты JUnit 5 для StandardCsvConverter в классе StandardCsvConverterTests.  Примечания:  1. Можно добавить новые интерфейсы/классы  2. Можно добавить новые зависимости библиотек |
| **public class** CsvConverter **implements** Converter {   */\*\*  \* Converts given {****@link*** *ConvertibleCollection} to CSV and outputs result as a text to the provided {****@link*** *OutputStream}  \*  \** ***@param collectionToConvert*** *collection to convert to CSV format  \** ***@param outputStream*** *output stream to write CSV conversion result as text to  \*/* @Override  **public void** convert(ConvertibleCollection collectionToConvert, OutputStream outputStream) {  *//* ***TODO: implement*** } } | **public class** CsvConverter **implements** Converter {   */\*\*  \* Конвертируйте {****@link*** *ConvertibleCollection} в CSV and выведите результат в виде текста {****@link*** *OutputStream}  \*  \** ***@param collectionToConvert*** *коллекции для конвертации в формат CSV   \** ***@param outputStream*** *поток вывода для записи результата преобразования CSV в виде текста*  *\*/* @Override  **public void** convert(ConvertibleCollection collectionToConvert, OutputStream outputStream) {  *//* ***TODO: implement*** } } |
| **public interface** ConvertibleCollection {   */\*\*  \* Gets headers of this convertible collection. All records of the collection are considered to have the  \* same headers  \*  \** ***@return*** *headers of the convertible collection  \*/* Collection<String> getHeaders();   */\*\*  \* Gets data records of this convertible collection  \*  \** ***@return*** *data records of the convertible collection  \*/* Iterable<ConvertibleMessage> getRecords();  } | **public interface** ConvertibleCollection {   */\*\*  \* Получает заголовки конвертируемой коллекции. Все записи коллекции имеют  \* одинаковые заголовки  \*  \** ***@return*** *заголовки конвертируемой коллекции  \*/* Collection<String> getHeaders();   */\*\*  \* Получает записи данных этой конвертируемой коллекции  \*  \** ***@return*** *записи данных конвертируемой коллекции  \*/* Iterable<ConvertibleMessage> getRecords();  } |
| **public interface** ConvertibleMessage {   */\*\*  \* Gets value of the provided element from this message  \*  \** ***@param elementId*** *id of the element to get value of  \** ***@return*** *value of the provided element  \*/* String getElement(String elementId);  } | **public interface** ConvertibleMessage {   */\*\*  \* Получает значение предоставленного элемента из этого сообщения  \*  \** ***@param elementId*** *id элемента для получения значения  \** ***@return*** *значение предоставленного элемента  \*/* String getElement(String elementId);  } |
| **public class** StandardCsvConverter **implements** StandardConverter {   **private final** CsvConverter **csvConverter**;   **public** StandardCsvConverter(CsvConverter csvConverter) {  **this**.**csvConverter** = csvConverter;  }  */\*\*  \* Converts given {****@link*** *List*<*Map*>*} to CSV and outputs result as a text to the provided {****@link*** *OutputStream}  \*  \** ***@param collectionToConvert*** *collection to convert to CSV format. All maps must have the same set of keys  \** ***@param outputStream*** *output stream to write CSV conversion result as text to  \*/* @Override  **public void** convert(List<Map<String, String>> collectionToConvert, OutputStream outputStream) {  *//* ***TODO: implement by using csvConverter*** } } | **public class** StandardCsvConverter **implements** StandardConverter {   **private final** CsvConverter **csvConverter**;   **public** StandardCsvConverter(CsvConverter csvConverter) {  **this**.**csvConverter** = csvConverter;  }  */\*\*  \* Конвертация {****@link*** *List*<*Map*>*} в CSV and выведите результат в виде текста {****@link*** *OutputStream}  \*  \** ***@param collectionToConvert*** *коллекции для конвертации в формат CSV. Все maps должны иметь одинаковый keys  \** ***@param outputStream*** *поток вывода для записи результата преобразования CSV в виде текста*  *\*/* @Override  **public void** convert(List<Map<String, String>> collectionToConvert, OutputStream outputStream) {  *//* ***TODO: implement by using csvConverter*** } } |