Recipe 1.2. How to create a data symbol manager ?

Table of contents

1. Problem

Once you have created a symbol, you want to couple it with the correct data content - it is the main task of the manager.

2. Solution

The data symbol manager handles the mapping of symbol with its content. Each manager have to provides implementation for method coming from the `DataSymbolRegister<T>` interface (definition of alphabet, mapping symbol to data, etc.).

Here is an example of implementation of the manager of `CardRankSymbol` that has already been defined in the creating symbol tutorial :

```
public class CardRankManager extends AbstractDataSymbolRegister<Integer> {

        private static CardRankManager INSTANCE = new CardRankManager();

        // link between symbols and card value
        private Map<Character, Integer> sym2ranks;

        private CardRankManager() {
                super(CardRankSymbol.getSymbolType());
        }

        public static final CardRankManager getInstance() {
                return INSTANCE;
        }

        @Override
        public Map<Character, Integer> createDataMap() {
                sym2ranks = new HashMap<Character, Integer>();

                for (int i = 2; i < 10; i++) {
                        addCard(String.valueOf(i).charAt(0), i - 1);
                }

                addCard('T', 9);
                addCard('J', 10);
                addCard('Q', 11);
                addCard('K', 12);
                addCard('A', 13);

                return sym2ranks;
        }

        private void addCard(final char symbol, final Integer rank) {
                if (!sym2ranks.containsKey(symbol)) {
                        sym2ranks.put(Character.valueOf(symbol), rank);
                } else {
                        throw new IllegalStateException(symbol + " already
exists.");
                }
        }

        @Override
        public Integer lookUpData(char name) {
                return sym2ranks.get(name);
        }

        @Override
        public void registerDataSymbol(Symbol<Integer> symbol, Integer
data) {
                if (!sym2data.containsKey(symbol)) {
                        sym2data.put(symbol, data);
                }
        }
}
```

Here is an example of usage:

```
// get the unique instance of card manager
CardRankManager manager = CardRankManager.getInstance();

// get the number of symbols
Assert.assertEquals(13, manager.getSymbolNumber());

// display [2, 3, 4, 5, 6, 7, 8, 9, T, J, Q, K, A]
System.out.println(manager.getLetters());

// display 10
System.out.println(manager.lookUpData('J'));

// display CardRankType.J
System.out.println(manager.lookUpSymbol('J'));
```

In the JPL we define an adapter class for mapping molecule named AbstractDataSymbolRegister<T extends Molecule> that give default implementations. AAManager is one of its subclass:

```
// here is an example of manager
AAManager aaManager = AAManager.getInstance();

// get the content
AminoAcid aa = aaManager.lookUpSymbol('A')

Assert.assertEquals("C3H5NO", aa.getFormula());
```

3. Discussion

4. See Also

See also how to create custom symbol and how to build sequence of symbols.