Java Mapping for Dictionaries

Here is the definition of our EmployeeMap once more:

Slice

```
dictionary<long, Employee> EmployeeMap;
```

As for sequences, the Java mapping does not create a separate named type for this definition. Instead, the dictionary is simply an instance of the generic type `java.util.Map<K, V>`, where *K* is the mapping of the key type and *V* is the mapping of the value type. In the example above, `EmployeeMap` is mapped to the Java type `java.util.Map<Long, Employee>`. The following code demonstrates how to allocate and use an instance of `EmployeeMap`:

Java

```
java.util.Map<Long, Employee> em = new java.util.HashMap<Long, Employee>();

Employee e = new Employee();
e.number = 31;
e.firstName = "James";
e.lastName = "Gosling";

em.put(e.number, e);
```

The type-safe nature of the mapping makes iterating over the dictionary quite convenient:

Java

```
for (java.util.Map.Entry<Long, Employee> i : em.entrySet()) {
    long num = i.getKey();
    Employee emp = i.getValue();
    System.out.println(emp.firstName + " was employee #" + num);
}
```

Alternate mappings for dictionary types are also possible.

See Also

- Dictionaries
- Java Mapping for Enumerations
- Java Mapping for Structures
- Java Mapping for Sequences
- Customizing the Java Mapping