



COLLECTION & MAP

ThS. Nguyễn Nghiệm
0913.745.789 - NghiemN@fpt.edu.vn



● Collection

- ✱ Phân cấp thừa kế
- ✱ List & ArrayList
- ✱ Set & HashSet
- ✱ Lớp tiện ích Collections

● Map

- ✱ Phân cấp thừa kế
- ✱ Map & HashMap
- ✱ Properties



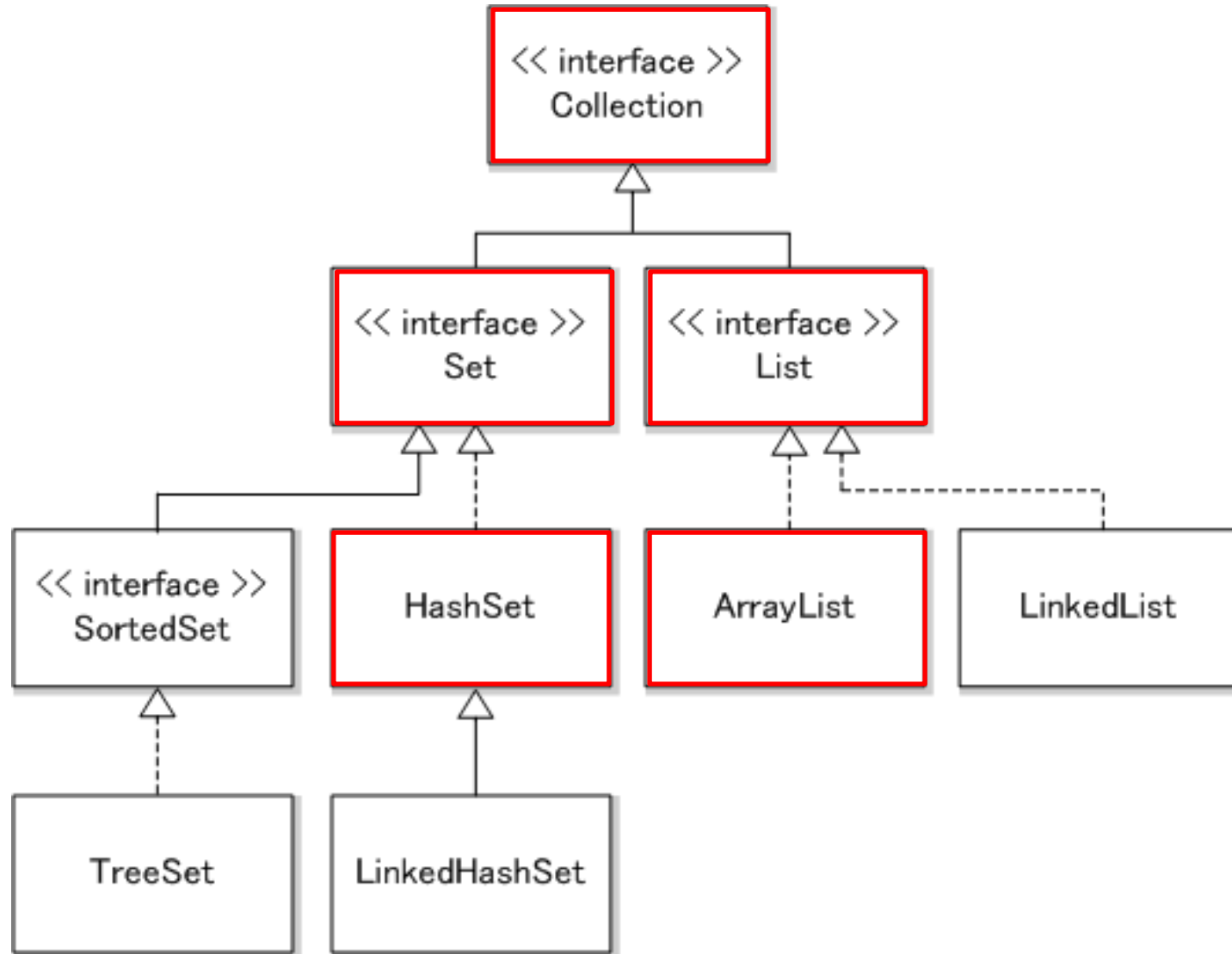


COLLECTION

- Collection là cấu trúc dữ liệu được sử dụng để nắm giữ nhiều phần tử.
 - ✱ Có thể thêm, xóa, cập nhật các phần tử.
 - ✱ Hợp, giao, trừ... các tập hợp
- Collection được chia làm 2 loại là List và Set
 - ✱ List là collection mà mỗi phần tử được phép xuất hiện nhiều lần và truy xuất bằng chỉ số
 - ✱ Set là collection mà mỗi phần tử chỉ được phép xuất hiện 1 lần và được phép truy xuất theo chỉ số



PHÂN CẤP THỪA KẾ TẬP





VÍ DỤ LIST & SET

```
List<Integer> list = new ArrayList<Integer>();  
list.add(1);  
list.add(2);  
list.add(2);  
System.out.print(list.toString());
```

[1, 2, 2]

```
Set<Integer> set = new HashSet<Integer>();  
set.add(100);  
set.add(200);  
set.add(200);  
System.out.print(set.toString());
```

[100, 200]



COLLECTION API

Phương thức	Mô tả
<code>boolean add(Object)</code>	Thêm vào
<code>addAll(Collection)</code>	Hợp 2 tập hợp
<code>boolean remove(Object)</code>	Xóa phần tử chỉ định
<code>removeAll(Collection)</code>	Hiệu 2 tập hợp
<code>retainAll(Collection)</code>	Giao 2 tập hợp
<code>boolean contains(Object)</code>	Kiểm tra sự tồn tại
<code>boolean containsAll(Collection)</code>	Kiểm tra sự tồn tại
<code>int size()</code>	Số phần tử
<code>boolean isEmpty()</code>	Rỗng hay không
<code>void clear()</code>	Xóa sạch
<code>toArray(T[])</code>	Chuyển đổi sang mảng



HỢP 2 TẬP HỢP

```
List<Integer> list = new ArrayList<Integer>();  
list.add(1);  
list.add(2);  
list.add(2);
```

```
Set<Integer> set = new HashSet<Integer>();  
set.add(100);  
set.add(200);  
set.add(200);
```

list.addAll(set)

?

```
set.addAll(list);  
System.out.print(set.toString());
```

[1,2,100,200]



LIST API

- Bên cạnh các phương thức thao tác tập hợp, List được bổ sung các phương thức làm việc với chỉ số

Phương thức	Mô tả
Object get(int index)	Truy xuất
Object set(int index, Object elem)	Cập nhật
void add(int index, Object elem)	Chèn thêm
Object remove(int index)	Xóa phần tử tại vị trí
int indexOf(Object elem)	Tìm vị trí từ đầu
int lastIndexOf(Object elem)	Tìm vị trí từ cuối



VÍ DỤ LIST

```
List<String> names = new ArrayList<>();  
  
names.add("Tuấn");  
names.add("Hạnh");  
names.add("Phương");  
names.add("Hằng");  
names.set(1, "Khanh");  
names.remove("Phương");  
  
System.out.println(names.toString());
```

[Tuấn, Khanh, Hằng]



DUYỆT LIST

```
List<String> names = new ArrayList<>();
```

```
for(int i=0;i<names.size();i++){  
    String name = names.get(i);  
    System.out.println(" >> Name: " + name);  
}
```

```
for(String name : names){  
    System.out.println(" >> Name: " + name);  
}
```

```
Iterator<String> iterator = names.iterator();  
while(iterator.hasNext()){  
    String name = iterator.next();  
    System.out.println(" >> Name: " + name);  
}
```



LỚP TIỆN ÍCH COLLECTIONS

Phương thức	Mô tả
<code>int binarySearch (List list, Object key)</code>	Tìm kiếm nhị phân
<code>void fill (List list, Object obj)</code>	Gán giá trị cho các phần tử
<code>void shuffle (List list)</code>	Hoán vị ngẫu nhiên
<code>void sort (List list)</code>	Sắp xếp tăng dần
<code>void reverse (List list)</code>	Đảo ngược
<code>void rotate (List list, int distance)</code>	Xoay vòng
<code>void swap(List list, int i, int j)</code>	Tráo đổi

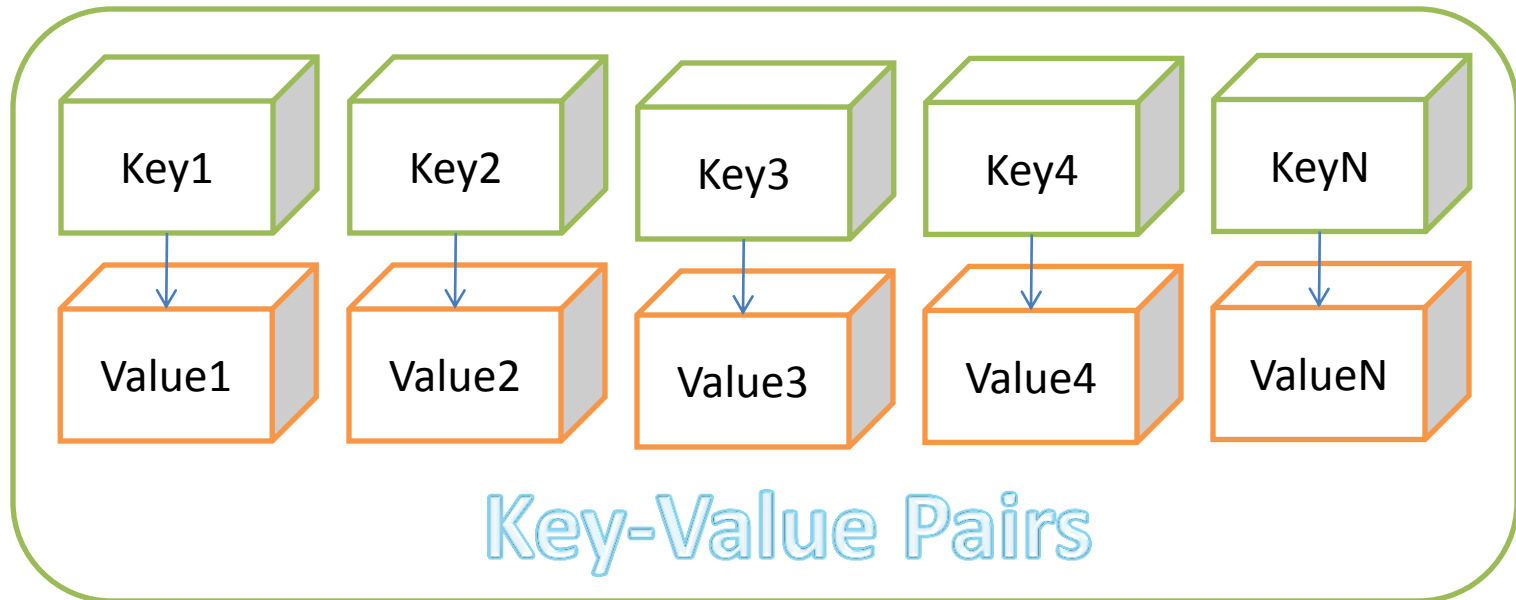


- Sử dụng **List<Product>** để lưu danh sách hàng hóa. Thông tin mỗi mặt hàng gồm mã, tên, giá, ngày SX. Thực hiện việc quản lý với các chức năng sau:
 - ✱ 1. Nhập danh sách hàng hóa
 - ✱ 2. Hiển thị lên bảng
 - ✱ 3. Tìm hàng hóa theo tên
 - ✱ 4. Tìm và sửa hàng
 - ✱ 5. Tìm và xóa
 - ✱ 8. Kết thúc



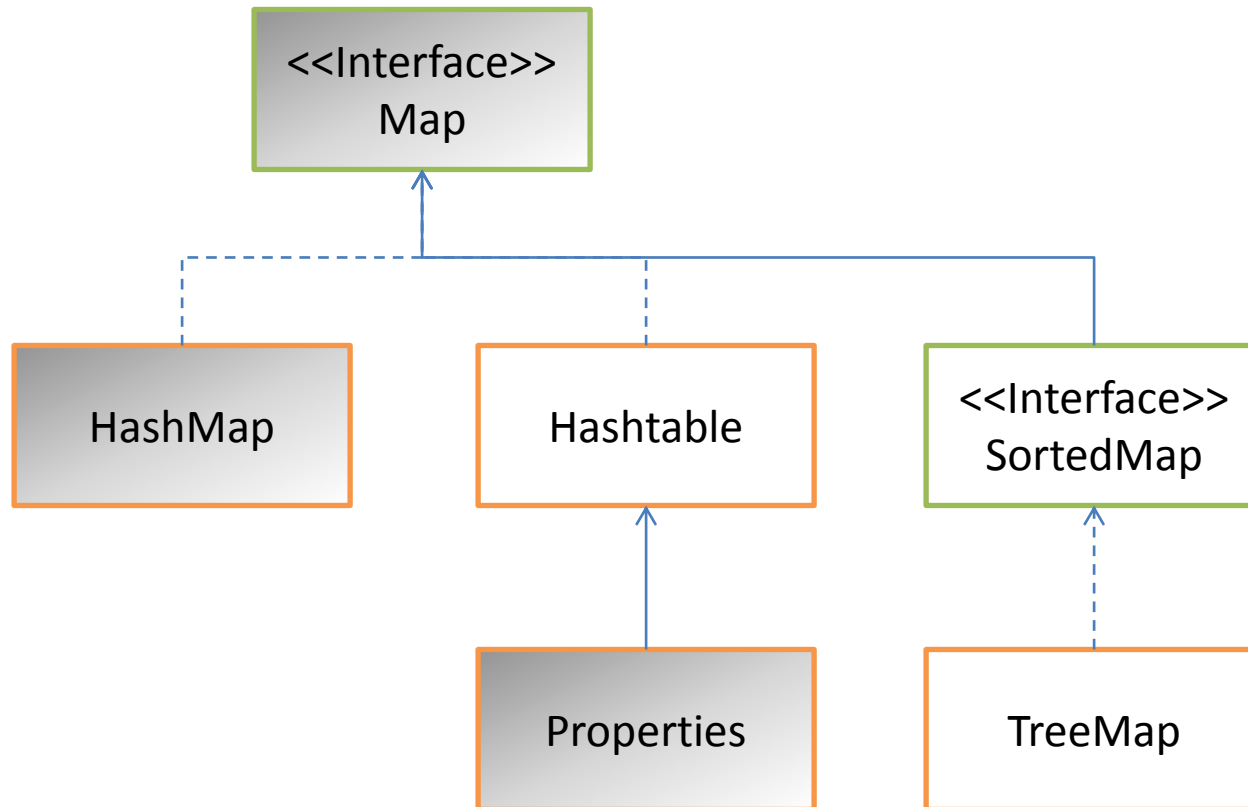
MAP

- Map là tập hợp các entry.
- Mỗi entry gồm key và value
- Sử dụng key để truy xuất giá trị của mỗi phần tử





PHÂN CẤP THỪA KẾ





VÍ DỤ MAP

```
// Khai báo tập hợp các ánh xạ giữa chuỗi và số thực
Map<String, Double> map = new HashMap<String, Double>();
// bổ sung 4 cặp vào tập hợp
map.put("Nokia", 500.0);
map.put("Samsung", 600.99);
map.put("Motorola", 399.99);
map.put("Sony Ericson", 400.50);
// cập nhật giá trị của phần tử có khóa là Samsung
map.put("Samsung", 555.55);
// chuyển sang chuỗi và xuất ra
System.out.print(map.toString());
```

{Motorola=399.990, Nokia=500.000, Sony
Ericson=400.500,Samsung=555.550}



MAP API

Phương thức	Mô tả
Object put (Object key, Object value)	Bổ sung hoặc cập nhật một phần tử
Object get (Object key)	Truy xuất một phần tử
Object remove (Object key)	Xóa một phần tử
boolean containsKey (Object key)	Kiểm tra sự tồn tại của key
int size ()	Lấy số lượng phần tử
boolean isEmpty ()	Kiểm tra có rỗng hay không
void clear ()	Xoá đi tất cả các ánh xạ.
Set keySet ()	Nhận danh sách khóa
Collection values ()	Nhận danh sách giá trị
Set entrySet ()	Nhận danh sách các cặp



DUYỆT MAP

```
Set<String> keys = map.keySet();  
for(String key: keys){  
    Double diem = map.get(key);  
}
```

```
for(Entry<String, Double> entry : map.entrySet()){  
    String ten = entry.getKey();  
    double diem = entry.getValue();  
}
```



PROPERTIES

- **Properties** là Map chuyên dụng với key và value là chuỗi.
- Cặp phương thức **getProperty()/setProperty()** thay cho **put()/get()** để làm việc với chuỗi
- Properties cũng cho phép đọc và lưu dữ liệu từ file thuộc tính hoặc file XML.



VÍ DỤ VỀ PROPERTIES

```
Properties props = new Properties();
```

```
props.setProperty("one", "Một");
```

```
props.setProperty("school", "Trường học");
```

```
props.setProperty("education", "Giáo dục");
```

```
props.setProperty("organization", "Tổ chức");
```

```
String mean1 = props.getProperty("education");
```

```
String mean2 = props.getProperty("two");
```

```
String mean3 = props.getProperty("two", "Chưa có");
```

```
System.out.println(props.toString());
```

Giáo dục

null

Chưa có

{organization=Tổ chức, school=Trường học,
one=Một, education=Giáo dục}

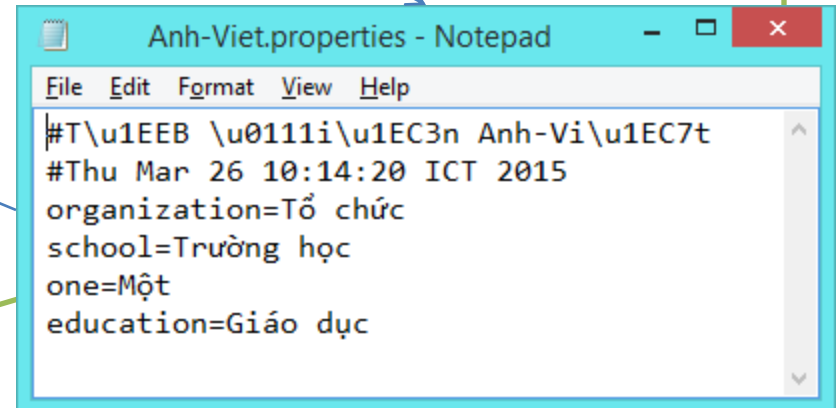


FILE PROPERTIES

```
try {  
    String fileName = "c:/temp/Anh-Viet.properties";  
    props.store(new FileWriter(fileName), "Tủ điển Anh-Việt");  
}  
catch (Exception e) {  
    e.printStackTrace();  
}  
  
try {  
    String fileName = "c:/temp/Anh-Viet.properties";  
    props.load(new FileReader(fileName));  
}  
catch (Exception e) {  
    e.printStackTrace();  
}
```

Store

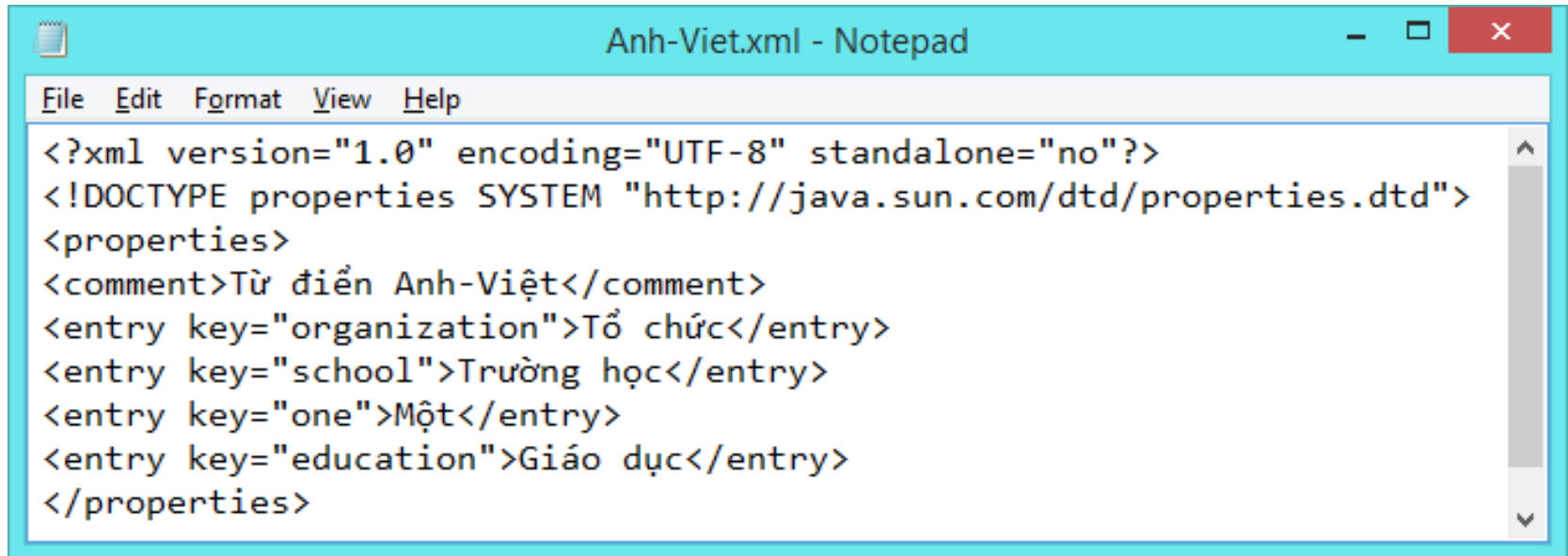
Load





FILE XML

- Cặp **loadFromXML()/storeToXML()** dùng để làm việc với file XML.



```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>
<comment>Tủ điển Anh-Việt</comment>
<entry key="organization">Tổ chức</entry>
<entry key="school">Trường học</entry>
<entry key="one">Một</entry>
<entry key="education">Giáo dục</entry>
</properties>
```



FILE XML

```
try {  
    String fileName = "c:/temp/Anh-Viet.xml";  
    props.storeToXML(new FileOutputStream(fileName), "Từ điển Anh-Việt");  
}  
catch (Exception e) {  
    e.printStackTrace();  
}  
  
try {  
    String fileName = "c:/temp/Anh-Viet.xml";  
    props.loadFromXML(new FileInputStream(fileName));  
}  
catch (Exception e) {  
    e.printStackTrace();  
}
```

StoreToXML

LoadFromXML

XML File



- Xây dựng từ điển online

- ✱ Tra cứu

- ✱ Thêm từ mới

- ✱ Hiển thị danh sách từ



● Collection

- ✱ Phân cấp thừa kế
- ✱ List & ArrayList
- ✱ Set & HashSet
- ✱ Lớp tiện ích Collections

● Map

- ✱ Phân cấp thừa kế
- ✱ Map & HashMap
- ✱ Properties