

Java: **Collection Klassen HashMap vs. Hashtable****Aufgabe:** Kennzeichenmapping

Wir wollen uns nun eine etwas andere Datenstruktur anschauen, die *HashMap*.

- Erstellen Sie hierfür eine neue Klasse *KFZMapping*. Um eine *HashMap* nutzen zu können, müssen Sie die Klasse importieren.

```
import java.util.HashMap;
```


- Sie sollten ein privates Objekt von *HashMap* in der neuen Klasse erzeugen.

```
private HashMap<Key , Value> map = new HashMap<Key , Value >();
```

Hinweis: <Key, Value> müssen durch die dementsprechende Klasse (n) ersetzt werden.

Mit einer *HashMap* können Schlüssel-Schloss-Paare gespeichert werden. In dieser Aufgabe sollen sowohl Schlüssel, als auch Schloss vom Typ *String* sein.

1. Erstellen Sie eine Methode `void insert(String kennzeichen, String landkreis)`, die ein Kennzeichen, wie zum Beispiel "S", und ein Landkreis, wie "Stuttgart" übergeben bekommt und das Schlüssel-Schloss-Paar in die *HashMap* einfügt.  
Nutzen Sie hierfür den `put(Key k, Value v)`-Befehl der Map.
2. Erstellen Sie eine Methode `void remove(String kennzeichen)`, die ein Kennzeichen-Landkreis-Paar aus der Map löscht.  
Nutzen Sie hierfür den `remove(Key k)`-Befehl der Map.
3. Erstellen Sie eine Methode `void printLandkreis(String kennzeichen)`, die den passenden Landkreis zum übergebenen Kennzeichen auf der Konsole ausgibt.  
Nutze hierfür den `get(Key k)`- Befehl der *HashMap*.
4. Fügen Sie zum Testen in der main-Methode ein paar Kennzeichen-Landkreis-Paare der Map hinzu. Drucken Sie ein Landkreis zu einem der Kennzeichen aus und löschen Sie ein anderes Kennzeichen aus der Map.  
Fügen Sie nun ein bereits vorhandenes Kennzeichen mit einem anderen Landkreis als bereits eingefügt der Map hinzu und geben Sie den Landkreis des Kennzeichens zurück.  
Was fällt ihnen auf?

PR1	WA-Java Collection Classes	
-----	----------------------------	---

**Aufgabe:** *HashMap* vs. *Hashtable* - Internetrecherche

Beantworten Sie folgende Fragen schriftlich:

1. Welche Gemeinsamkeiten haben die Klassen *HashMap* und *HashTable*
2. Wie unterscheiden sich die beiden Klassen, wenn man Sie zur Laufzeit in verschiedenen parallelen Prozessen(Threads) laufen lässt?
3. Wie unterscheiden sich die Klassen wenn man versucht als Key (Schlüssel) einen *null* Wert zu setzen und wie bei *null* Werten als Value (Schloss)?

Antworten: