Java HashMap

Java HashMap

In the ArrayList chapter, you learned that Arrays store items as an ordered collection, and you have to access them with an index number (int type). A HashMap however, store items in "**key/value**" pairs, and you can access them by an index of another type (e.g. a String).

One object is used as a key (index) to another object (value). It can store different types: String keys and Integer values, or the same type, like: String keys and String values:

Example

Create a HashMap object called capitalCities that will store String keys and String values:

```
import java.util.HashMap; // import the HashMap class

HashMap<String, String> capitalCities = new HashMap<String, String>();
```

Add Items

The HashMap class has many useful methods. For example, to add items to it, use the put() method:

Example

```
// Import the HashMap class
-------------------------------------------------------------------------------
      import java.util.HashMap;

      public class Main {
        public static void main(String[] args) {
          // Create a HashMap object called capitalCities
          HashMap<String, String> capitalCities = new HashMap<String, String>();

          // Add keys and values (Country, City)
          capitalCities.put("England", "London");
          capitalCities.put("Germany", "Berlin");
          capitalCities.put("Norway", "Oslo");
          capitalCities.put("USA", "Washington DC");
          System.out.println(capitalCities);
        }
      }
-------------------------------------------------------------------------------
```

Access an Item

To access a value in the HashMap, use the get() method and refer to its key:

Example

```
capitalCities.get("England");
```

```
import java.util.HashMap;

public class Main {
  public static void main(String[] args) {
    HashMap<String, String> capitalCities = new HashMap<String, String>();
    capitalCities.put("England", "London");
    capitalCities.put("Germany", "Berlin");
    capitalCities.put("Norway", "Oslo");
    capitalCities.put("USA", "Washington DC");
    System.out.println(capitalCities.get("England"));
  }
}
```

Remove an Item

To remove an item, use the remove() method and refer to the key:

Example

```
capitalCities.remove("England");
```

```
import java.util.HashMap;

public class Main {
  public static void main(String[] args) {
    HashMap<String, String> capitalCities = new HashMap<String, String>();
    capitalCities.put("England", "London");
    capitalCities.put("Germany", "Berlin");
    capitalCities.put("Norway", "Oslo");
    capitalCities.put("USA", "Washington DC");
    capitalCities.remove("England");
    System.out.println(capitalCities);
  }
}
```

To remove all items, use the clear() method:

Example

```
capitalCities.clear();
```

```
import java.util.HashMap;

public class Main {
  public static void main(String[] args) {
    HashMap<String, String> capitalCities = new HashMap<String, String>();
    capitalCities.put("England", "London");
    capitalCities.put("Germany", "Berlin");
    capitalCities.put("Norway", "Oslo");
    capitalCities.put("USA", "Washington DC");
    capitalCities.clear();
    System.out.println(capitalCities);
  }
}
```

HashMap Size

To find out how many items there are, use the size() method:

Example

```
capitalCities.size();
```

```
import java.util.HashMap;

public class Main {
  public static void main(String[] args) {
    HashMap<String, String> capitalCities = new HashMap<String, String>();
    capitalCities.put("England", "London");
    capitalCities.put("Germany", "Berlin");
    capitalCities.put("Norway", "Oslo");
    capitalCities.put("USA", "Washington DC");
    System.out.println(capitalCities.size());
  }
}
```

Loop Through a HashMap

Loop through the items of a HashMap with a **for-each** loop.
Note: Use the keySet() method if you only want the keys, and use the values() method if you only want the values:

Example

```
// Print keys
for (String i : capitalCities.keySet()) {
  System.out.println(i);
}
```

```
import java.util.HashMap;

public class Main {
  public static void main(String[] args) {
    HashMap<String, String> capitalCities = new HashMap<String, String>();
    capitalCities.put("England", "London");
    capitalCities.put("Germany", "Berlin");
    capitalCities.put("Norway", "Oslo");
    capitalCities.put("USA", "Washington DC");

    for (String i : capitalCities.keySet()) {
      System.out.println(i);
    }
  }
}
```

Example

```
// Print values
for (String i : capitalCities.values()) {
  System.out.println(i);
}
```

```
import java.util.HashMap;

public class Main {
  public static void main(String[] args) {
    HashMap<String, String> capitalCities = new HashMap<String, String>();
    capitalCities.put("England", "London");
    capitalCities.put("Germany", "Berlin");
    capitalCities.put("Norway", "Oslo");
    capitalCities.put("USA", "Washington DC");

    for (String i : capitalCities.values()) {
```

```
            System.out.println(i);
          }
        }
      }
```

Example

```
// Print keys and values

for (String i : capitalCities.keySet()) {

  System.out.println("key: " + i + " value: " + capitalCities.get(i));

}
```

```
      import java.util.HashMap;

      public class Main {
        public static void main(String[] args) {
          HashMap<String, String> capitalCities = new HashMap<String, String>();
          capitalCities.put("England", "London");
          capitalCities.put("Germany", "Berlin");
          capitalCities.put("Norway", "Oslo");
          capitalCities.put("USA", "Washington DC");

          for (String i : capitalCities.keySet()) {
            System.out.println("key: " + i + " value: " + capitalCities.get(i));
          }
        }
      }
```

Other Types

Keys and values in a HashMap are actually objects. In the examples above, we used objects of type "String". Remember that a String in Java is an object (not a primitive type). To use other types, such as int, you must specify an equivalent wrapper class: `Integer`. For other primitive types, use: `Boolean` for boolean, `Character` for char, `Double` for double, etc:

Example

Create a HashMap object called people that will store String keys and Integer values:

```
// Import the HashMap class
-------------------------------------------------------------------------------
      import java.util.HashMap;

      public class Main {
        public static void main(String[] args) {

          // Create a HashMap object called people
          HashMap<String, Integer> people = new HashMap<String, Integer>();


          // Add keys and values (Name, Age)
          people.put("John", 32);
          people.put("Steve", 30);
          people.put("Angie", 33);

          for (String i : people.keySet()) {
            System.out.println("key: " + i + " value: " + people.get(i));
          }
        }
      }
-------------------------------------------------------------------------------
```