**MAPS**

Diagram

Description automatically generated

Biz herhangi bir SET yapisinda verileri duzensiz olarak tutabiliriz

Yada bir LIST icerisinde ekleme sirasina göre verilerimizi duzenli olarak tutabiliyoruz.

Fakat MAP yapisinda biz, datalari bu ikisinden farkli olarak key ve value seklinde tutuyoruz.

Reel projelerde kullanilan database yapisina en uygun Java objesidir.

Map ile bir objeye ait farkli bilgileri kategorilerine gore ayirip depolayabiliriz

Maplerde unique key ve degisken olacabilecek valuelarimiz vardir.

Key’ler unique olmalidir.

Maps key – value pairs kullanir ( anahtar –deger(ler) ).

Ve biz bir yerden desimeyen keyimizi cagirdigimizda bize valuesini tertemiz getirir

Bizim Map interface yardimiyla kullanabilcegimiz iki stardart yapi var :

1-HashMap

2-TreeMap

HashMap yapisinda eklenme siralamasi diye birsey yok. Kendi algoritmasi var ve kendince bir duzen olusturuyor

Fakat TreeMapte hic olmazsa istememiz durumunda anahtarlara gore bir siralama yapabiliriz. Ama tabi bunda eklenme sirasi yine onemli degil.

Yani ben rastgele bir liste olusturuyorum ve bu liste 1\_Hashmapte javanin kendi algoritmasina gore kendince bir siralamayla konsolda görülüyor

2-TreeMapte ise istersem buyukten kucuge, kucukten buyuge bir siralama olusturtabilirim. Istemezsemde canim sagolsun :)

HashMap ve TreeMap synchronized ve thread-safe degildir.

Map'ler arasindan synchronized ve thread-safe olan bir map'e ihtiyac vardir.

Bu yuzden Hashtable olusturulmustur.

Hashtable TRICK'i :Hashtable ne key icin ne de value icin "null" kullanamaz.

Hashtable elemanlari rastgele siralar.

**Kullanim Amaci :**

1-Elinizdeki datalarin bir referansi varsa Map kullanmak isteyebilirsiniz

2-Yada kendince bir siralamaya tabi bir dökümani Maplerle olusturmak isteyebiliriz

**MAP METHODLARI.**

**put() :** HashMaplerde Listteki gibi add olayi yok Bunun yerine PUT var.Eleman eklemek icin put methodu kullanilir... key, value seklinde data eklenir..Text

Description automatically generated

**containsKey() :**Bir anahtar in varligini kontrol etmek icin sorgulamak icin true/false döner

**containsValue() :**Bir degerin varligini kontrol etmek icin sorgulamak icin true/false döner Graphical user interface, text

Description automatically generated

**EntrySet() :** Eklenen datalarin listesini görürüz. Map’deki entry’leri bir Set olarak verir.

Entry : Map’de her bir elemani olusturan key-value ikilisidir

2. sysoda sadece anahtarlari görürüm. sadece anahtarlari, set yapisiyla olusturup aliyoruz

Graphical user interface, text

Description automatically generated

**remove() :** veriyi silmek icin kullanilir

System.*out*.println(list.remove(456789456)); // sildi ve sildigini kanit olarak console da gösterdi

System.*out*.println(list); //{1236987=filo, 123456789=Halo}

**size() :** hashmap icerisindeki data sayisini entry sayisini bize verir

System.*out*.println(list.size()); // 2

**equals(map);** Map’deki tum elemanlari karsilastirir. Hepsi ayni ise true farkli olan varsa false dondurur

**get(key);** istenen key degeri Map’de varsa o key’e ait value’yu, map’de yoksa null doner.

**getOrDefault(key,defaultDeger);** istenen key degeri Map’de varsa o key’e ait value’yu, key map’te yoksa default degeri doner.

**putAll(map);** verilen map’deki tum elemanlari bizim map’imize ekler, tekrarlanan eleman varsa uzerine yazar

**compute(key, (key,value)->yeniDeger);** verilen map’deki istenen key degerine sahip elemanin value’sunu gunceller key map’te yoksa ekler

**ComputeIfPresent(key, (key,value)->yeniDeger);** istenen key degeri Map’de varsa o key’e ait value’yu gunceller, map’de yoksa birsey yapmaz

**ComputeIfAbsent(key, k ->yeniDeger);** istenen key degeri map’de yoksa o key’i ve value’yu ekler, map’de varsa birsey yapmaz

**putIfAbsent(key, value);** verilen key map’de yoksa ekler.

Diagram

Description automatically generated

Timeline

Description automatically generated