44.video

1. Interface : bildigimiz class yapisindan farklidir, interface bir class degildir.Interface’lerin

constructor’lari yoktur ve obje olusturulamaz.

1. Java da full abstraction yapmak icin interface kullaniriz, interface icerisinde concrete method

bulunmaz ve abstract class’lara gore yavastir.

1. Interface Child class’larin bulundurmak zorunda olduklari method’lari belirleyen bir to-do list gibidir
2. Concrete class’lardan interface’i inherit etmek icin extends keyword yerine implements keyword kullanilir.
3. Eger bir interface’den baska bir interface’i inherit edeceksek extends keyword kullaniriz
4. Concrete class’lar birden fazla concrete class’i extends ile inherit edemezler ancak implements ile bir class birden fazla interface’I inherit edebilir.
5. Interface icerisindeki tum variable’lar abstract,public ve final’dir, final olduklari icin de olustururken deger atamak zorunludur.
6. Bir interface icerisinde body’si olan method olusturulabilir mi ?

- Java interface icerisinde body’si olan method olusturabilmemiz icin 2 alternatif gelistirmistir

A- default keyword kullanilabilir. ( burada kullanilan default access modifier olan degildir)

B- static keyword kullanilabilir.

aralarindaki fark default keyword ile olusturulan method’a diger class’lardan obje ile ulasabilirken, static keyword ile olusturulana Interface ismi ile ulasabiliriz.

Socrative

Biz bugune kadar birden fazla nesne barindiran data turleri olarak list ve array gorduk, ikisinde de index vardir. Yani biz bir list veya arrayin 3. elemanina index araciligiyla ulasabiliriz. Ama list ve array disinda da birden cok nesne barindiran data turleri vardir. bazilarinda index vardir bazilarinda ise index yoktur.

Collectionda karsimiza indexi olmayan turler de cikacak, bu durumlarda elemanlar arasinda gezinmek bir sorun olur.

**List <Integer> liste = new ArrayList<>();**

**liste.add(5);**

**liste.add(3);**

**liste.add(7);**

**liste.add(1);**

**liste.add(9);**

**liste.add(8);**

**liste.add(2);**

**System.out.println(liste);**

**// index kullanmadan tum elemanlari 3 eklenmis olarak yazdirin**

**for (Integer each : liste) {**

**System.out.print(each+3 + " ");**

**}**  **// 8 6 10 4 12 11 5**

**System.out.println("");**

**System.out.println(liste); //[5, 3, 7, 1, 9, 8, 2]**

**for (Integer each : liste) {**

**each+=3;**

**System.out.print(each + " ");}**

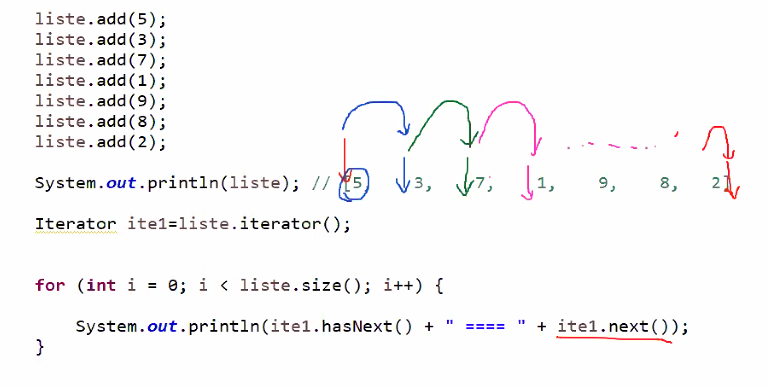
**System.out.println("");**

**System.out.println(liste);**

**// foreach loop ile index kullanmadan tum elementlere ulasabiliyoruz ama**

**// kalici degisiklik yapamiyoruz**

Iterator in mantigi imlec mantigi gibidir, siz bir iterator olusturdugunuzda ortalikta olusturamazsiniz, iterator I bir collections uyesi icin olusturabilirsiniz.



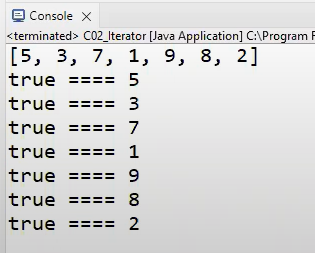
Iterator bir amac icin olusturulur. Bu class icin amac bu listedir. Biz iterator I olusturdugumuzda iterator listenin basina gelir ve durur. Iteratorda 3 method vardir

hasNext();----> true yada false doner

next();

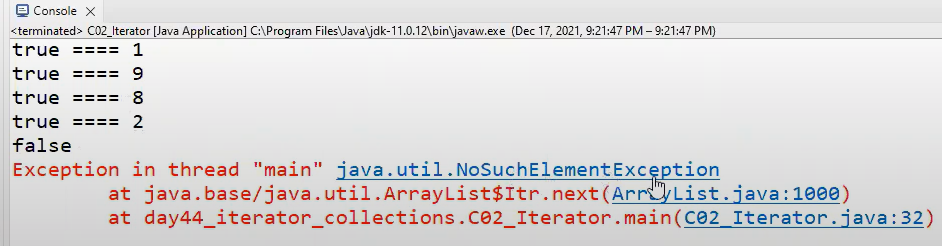
remove();

Bir for loop olusturalim …



Bu islemden sonra cursor 2den sonrada kalir. Biz bu andan sonra hasNext(); methodu kullanirsan bize false doner.

Biz yaninda eleman olmadigini bildigimiz halde ite.next() dersek java bize rte firlatir.



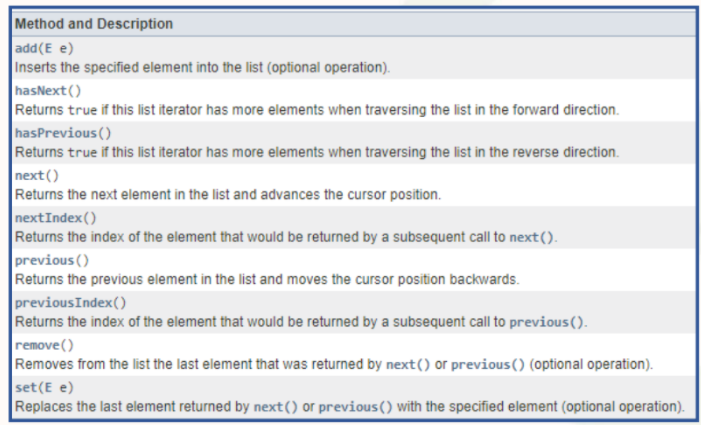
Yani iterator sirasiyla, index e bagli olmaksizin elementleri gezer. For each mantigi gibidir ama bu sirali gider.

Simdi beraber C03-04\_Iterator classlarina bakalim:

Iterator ----> interface dir.

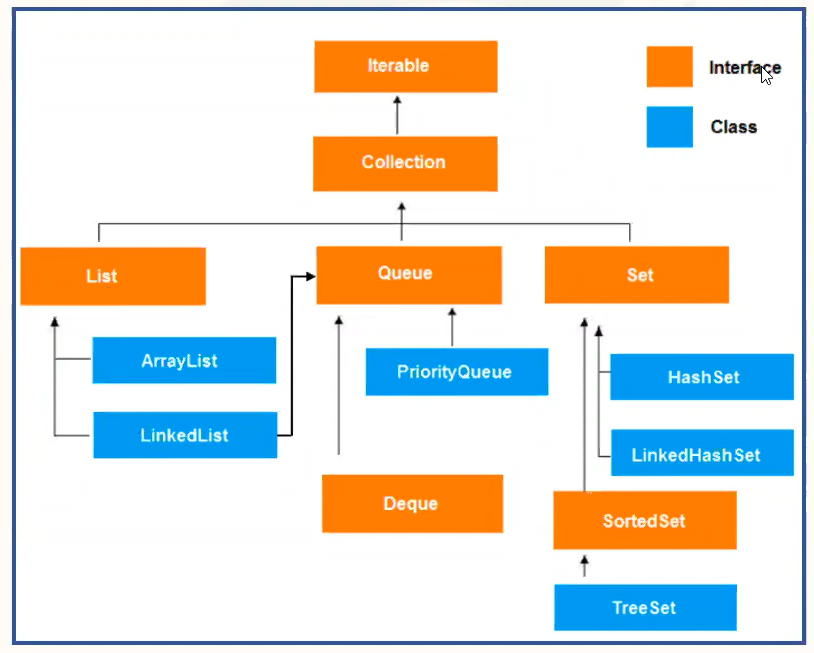
Icinde bodysi olmayan hasHext() next(); ve remove(); methodlari vardir (remove un bodysi var ama sadece exception firlatmak icin )

Bunun altinda bir child olmali, bunun childi ListIterator vardir ve daha fazla methodu vardir.



5.class a bakalim...

Collections



Collections : nesnelerden oluşan bir topluluğu bir arada tutan yapılardir.

Ornegin list bir interfaceymis parentlari da hep interface, biz iteratordaki interfacesi altindaki collectionda veya listte somutlastiramadik. Ancak array listten bir constractor olusturdugumuzda somutlastirabildik.

Bizim asil kullanacagimiz classlar mavi olanlardir. Sari olanlar uyulmasi gereken kurallari belirler.

Bizim icin bu tablodaki yapiyi anlamak onemli. Burada bizim icin onemli olan 5 kelime var

1. Set (kume)---🡪tekrarsiz
2. Queue (sira)
3. Linked (bagli)
4. Tree (siralama)--🡪dogal sirali
5. Hash (ozel bir siralama yontemi)