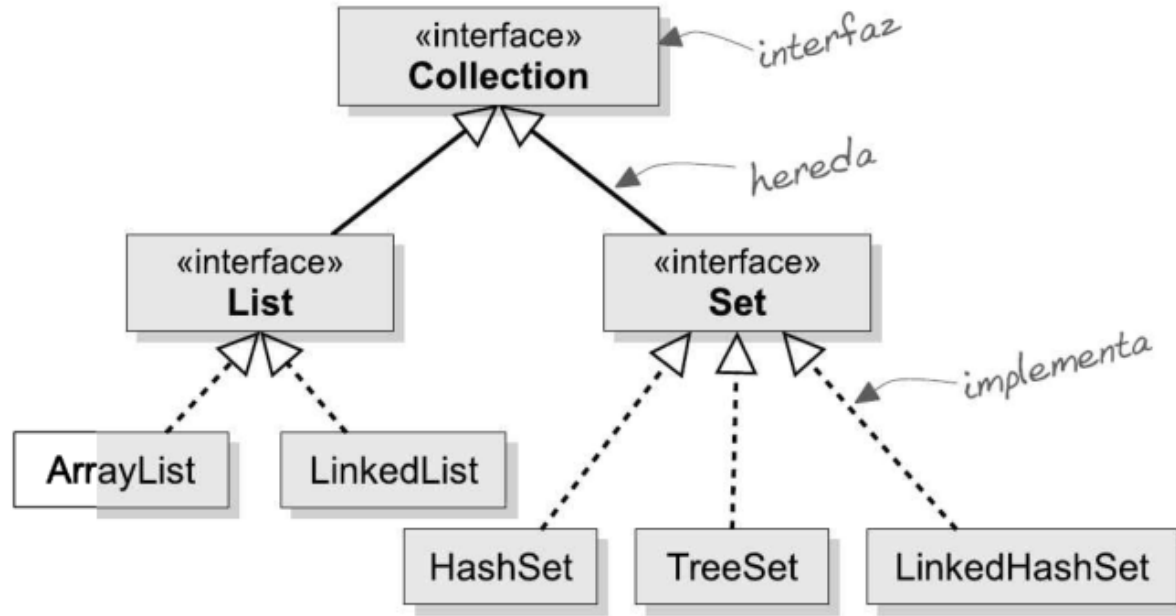




Collections

- Estructuras dinámicas -

Collection



Collection

```
ArrayList <E> lista = new ArrayList<>();
```

```
ArrayList <Cliente> lista = new ArrayList<Cliente>();
```

```
ArrayList <Cliente> lista = new ArrayList<>();
```

Collection

boolean add(E elem)

boolean remove(Object ob)

void clear()

int size()

boolean isEmpty()

boolean contains(Object ob)

String toString()

boolean hasNext()

Object next()

void remove()

BufferedReader

```
boolean containsAll(Collection <?> c)
boolean addAll(Collection <? extends E> c)
boolean removeAll(Collection <?> c)
boolean retainAll(Collection <?> c)

Object[] toArray()
<T>T[] toArray(T[] t)

E get(int indice)
E set(int indice, E elem)
void add(int indice, E elem)
boolean addAll(int indice, Collection<? extends E> c)
E remove(int indice)
```

BufferedReader

```
int indexOf(Object ob)  
int lastIndexOf(Object ob)  
boolean equals(Object otraLista)
```

FileWriter

```
int size()  
boolean isEmpty()  
boolean contains(Object element)  
boolean add(E element)  
boolean remove(Object element)  
Iterator<E> iterator()  
boolean containsAll(Collection<?> c)  
boolean addAll(Collection <? extends E> c)  
boolean removeAll(Collection<?> c)  
boolean retainAll(Collection<?> c)  
void clear()  
Object[] toArray()  
<T> T[] toArray(T[])
```

FileWriter

```
try {  
    FileWriter in = new FileWriter("Main.java")  
} catch (IOException ex) {  
    System.out.println(ex.getMessage());  
}
```


BufferedWriter

```
BufferedWriter in = new BufferedWriter(new FileWriter("prueba.txt"));
```

BufferedWriter

```
try {  
    BufferedWriter out = new BufferedWriter(new FileWriter("Machado.txt"));  
    String cad = "Mi infancia son recuerdos de un patio de Sevilla";  
    for (int i = 0; i < cad.length(); i++) {  
        out.write(cad.charAt(i));  
    }  
    cad = "y un huerto claro donde madura el limonero";  
    out.newLine();  
    out.write(cad);  
    out.close();  
} catch (IOException ex) {  
    System.out.println(ex.getMessage());  
}
```





Ficheros binarios

- Serialización -

FileOutputStream

```
FileOutputStream(String nombre_archivo)  
FileOutputStream archivo = new FileOutputStream("enteros.dat");  
ObjectOutputStream out = new ObjectOutputStream(archivo);
```

Serializable

```
class miClase implements Serializable {  
... // cuerpo de la clase  
}
```

OutputStream


void writeBoolean (boolean b)	escribe un valor boolean en el flujo
void writeChar(int c)	escribe el valor char del valor entero que se pasa
void writeInt(int n)	escribe un entero
void writeLong(long n)	escribe un entero largo
void writeDouble(double d)	escribe un double
void writeObject(Object o)	escribe un objeto <i>Serializable</i>

Ejemplo

```
int [] t = new int[10];
for (int i = 0; i < t.length(); i++) {t[i] = i;}

try {
    ObjectOutputStream flujo_salida;

    flujo_salida = new ObjectOutputStream(new
    FileOutputStream("datos.dat"))
    for (int i = 0; i < t.length(); i++) {
        flujo_salida.writeInt(t[i]);
    }
    flujo_salida.close();
} catch (IOException e) {
    System.out.println("¡Error al escribir el fichero!");
}
```



FileInputStream

```
FileInputStream(String nombre_archivo)  
FileInputStream archivo = new FileInputStream("enteros.dat");  
ObjectInputStream in = new ObjectInputStream(archivo);
```


Ejemplo

```
int [] tabla = new int[10];  
for (int i = 0; i < 10; i++) {  
    tabla[i] = flujoEntrada.readInt();  
}
```

```
try {  
    String cadena = (String) flujoEntrada.readObject();  
} catch (ClassNotFoundException ex) {  
    System.out.println(ex.getMessage());  
}
```

ObjectInputStream

boolean readBoolean ()	lee un valor boolean en el flujo
char readChar()	lee un valor char del valor entero que se pasa
int readInt()	lee un entero
long readLong()	lee un entero largo
double readDouble()	lee un double
Object readObject()	lee un objeto <i>Serializable</i>

ClassNotFoundException

```
try {  
    tabla = (int[]) in.readObject();  
} catch (ClassNotFoundException ex) {  
    System.out.println(ex.getMessage());  
}
```

“Si hemos guardado la tabla de enteros con **writeObject**, tendremos que leerla con **readObject**, que nos devolverá un **Object**, por lo que tendremos que *castearlo* a array de enteros **int[]**”

EOFException

```
try {  
    while (true) {  
        System.out.println(in.readInt());  
    }  
} catch (EOFException ex) {  
    System.out.println("Fin de fichero");  
}
```

“Como no sabemos el número de enteros que se han guardado, vamos leyendo en un bucle infinito hasta que salta la excepción **EOFException** (Excepción de Fin de Fichero)”



Cierre de flujos

```
OutputStream out = null;
try {
    out = new OutputStream(new FileOutputStream("archivo.dat"));
    ... // sentencias que manipulan el fichero
} catch (IOException ex) {
    System.out.println(ex.getMessage());
} finally {
    try {
        if (out != null)
            out.close();
    } catch (IOException ex) {
        System.out.println(ex.getMessage());
    }
}
```

Cierre de flujos

```
try (OutputStream out = new OutputStream(  
    new FileOutputStream("archivo.dat")) {  
    ... // sentencias que manipulan el fichero  
} catch (IOException ex) {  
    System.out.println(ex.getMessage());  
}
```