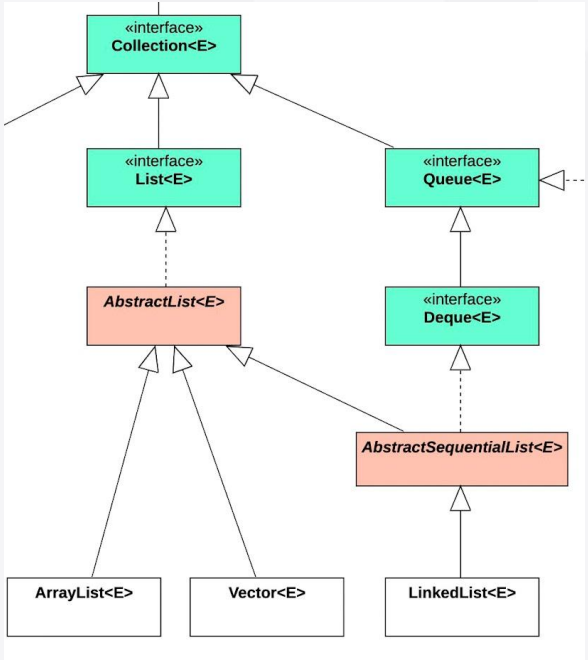
LIST

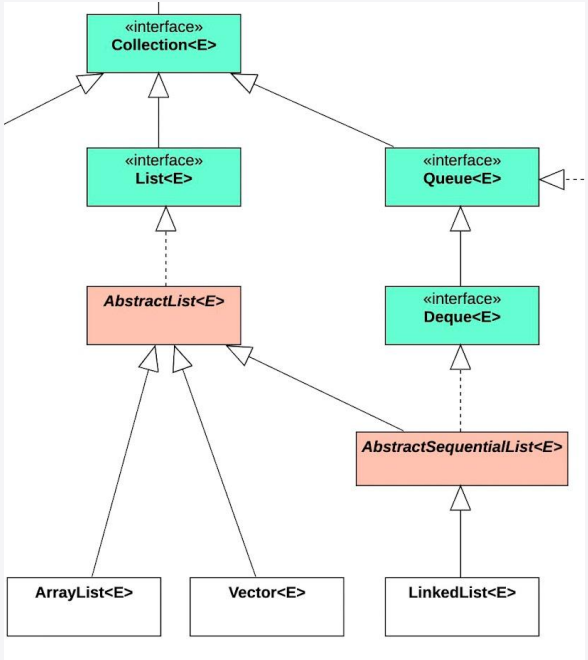
COLECCIONES

LIST<E>

- Interfaz
- Hereda de Collection<E>
- Añade:
  - Acceso posicional (*get*, *set*)
  - Búsqueda (indexOf, lastIndexOf)
  - Iteración extendida
  - Operaciones sobre una sublista



ALGUNAS IMPLEMENTACIONES DE LIST<E>



Más usual

Sincronizada, pero con métodos *legacy*. Menos recomendable

Apropiada para usarla como Queue (cola) o Deque (cola doble, que puede ser usada como pila)

CREACIÓN DE UNA LISTA

- Java 8: Arrays.asList(...)
  - Crea una lista de **tamaño fijo** (no se pueden añadir elementos).
  - Los elementos se puede modificar y sustituir.
- Java >= 9: List.of(...)
  - Crea una lista **no modificable** con los elementos que se proporcionan como argumentos.
  - No se pueden añadir, borrar o sustituir.
  - Si los elementos son mutables, se pueden modificar.

ARRAYLIST: LA COLECCIÓN MÁS USADA

- Implementación más adecuada en la mayoría de las ocasiones.
- Acceso e inserción eficiente (en media, *O(1)*)
- Ocupa menos espacio que otras implementaciones.
- No sincronizada (ojo si se usa concurrencia).

```
/*
 * Lista mutable de tamaño "indefinido"
 */
List<Integer> lista3 = new ArrayList<>();
```

RECORRER UNA LISTA

- Bucles for, for-each
  - No se permite modificar la lista recorrida (sí los elementos contenidos).
- Acceso posicional.
  - método get

RECORRER UNA LISTA

- Iterador
  - Hacia adelante: *Iterator* (*hasNext*, *next*)
  - Hacia atrás: *ListIterator* (*hasPrevious*, *previous*)
  - Opciones de modificación (*remove*)

```
for(Iterator<Integer> it = lista4.iterator(); it.hasNext(); ) {
    int n = it.next();
    if (n % 10 == 0 || n % 10 == 5)
        it.remove();
}

System.out.println(lista4);
```

OPERACIONES CON LIST<E>

- Búsqueda: Primer índice donde se encuentra un elemento
  - *indexOf* : desde el inicio
  - *lastIndexOf* : desde el final
- Contenido: Buscar si un elemento está contenido
  - *contains* : un elemento
  - *containsAll* : todos los elementos de una colección

OPERACIONES CON LIST<E>

- Tamaño y contenido
  - *size*: nº de elementos
  - *isEmpty*: true si size == 0
- Edición
  - *add*, *addAll*: añadir elementos.
  - *set*: sustituir un elemento por otro (posicionalmente)
  - *remove*, *removeAll*: eliminar uno o varios
  - *clear*: vaciar la colección (size = 0)

SUBLISTAS Y CONVERSIÓN A ARRAYS

- Sublistas: *subList*
  - Porción de una lista.
  - No almacena una copia de los elementos. Referencia a los elementos de la lista.
  - Cualquier operación de cambio se propaga: original → sublista, sublista → original.
- Arrays: *toArray*
  - Permite convertir el List<E> en un Object[] o en un E[ ]
  - Ojo con tipos primitivos :/