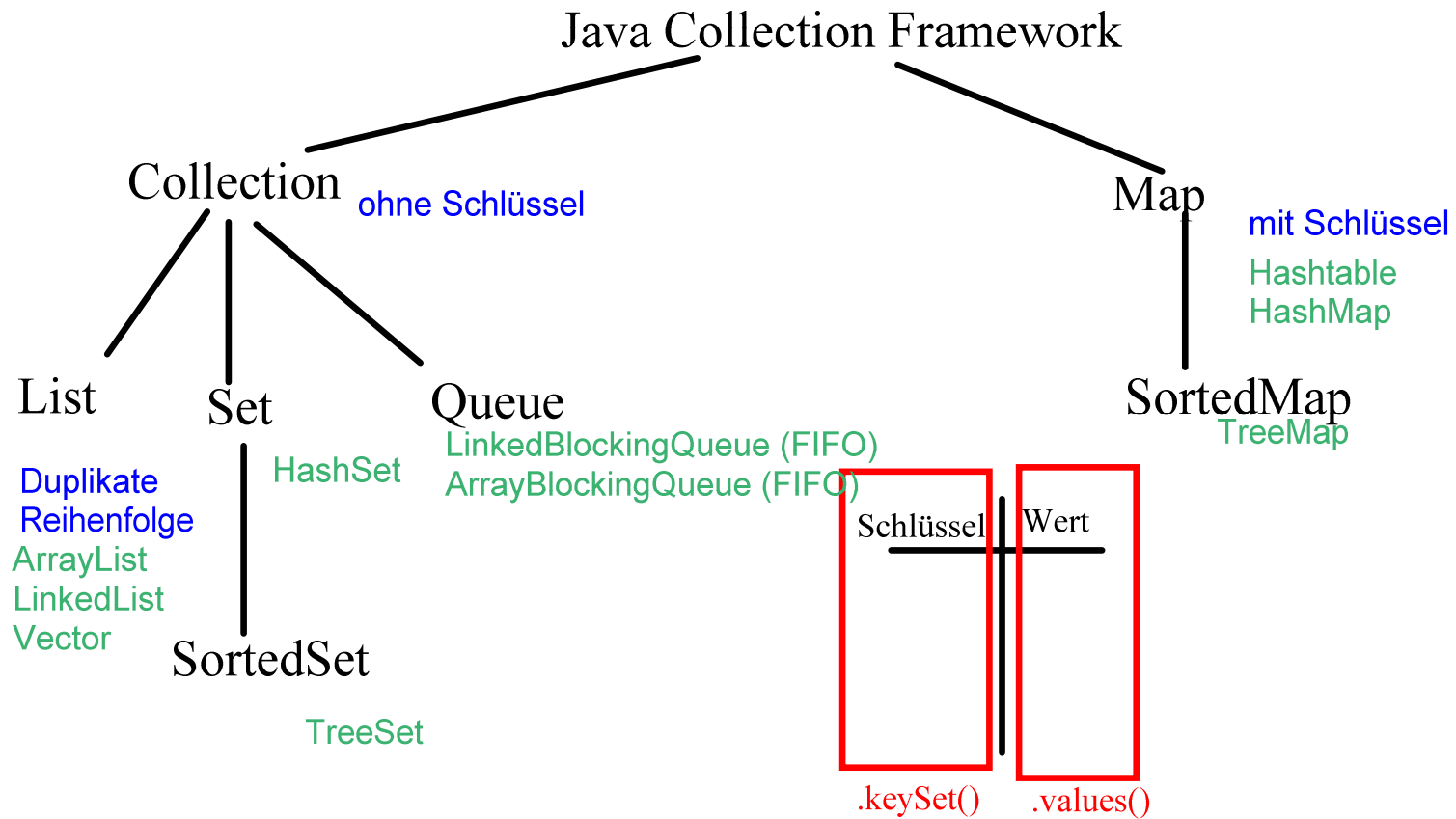


Montagsübungsstunde (einmalig) auf Mittwoch, 16.12.2009, 3.
Stunde verschoben



Abbilden von 1:n-Beziehungen (hier 1 Patient hat n Krankheiten)

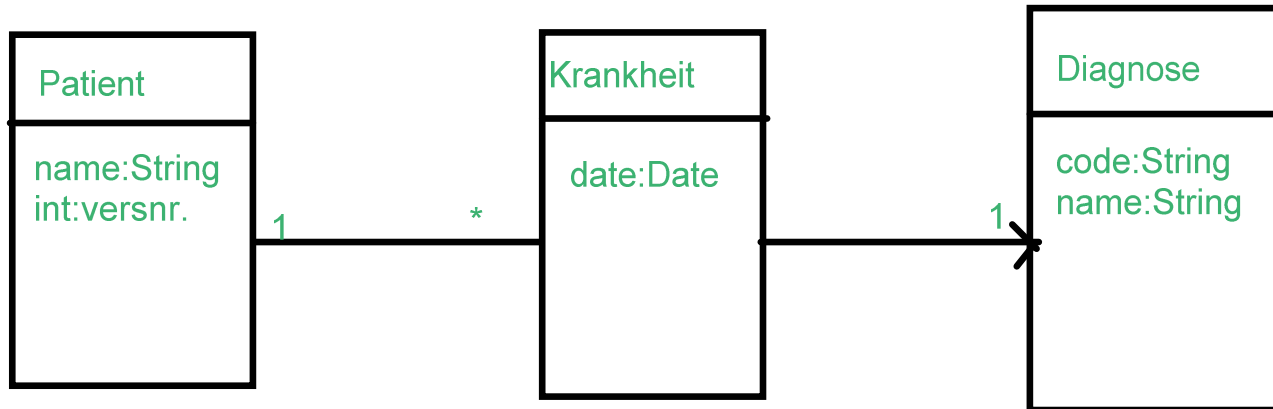
In spitzen Klammern:
Datentyp der Werte im Container

```
public class Patient {  
    public String name;  
    public int versicherungsnummer;  
    private Collection<Krankheit> krankheiten = new TreeSet<Krankheit>();  
    public Patient(String name, int versicherungsnummer) {  
        this.name = name;  
        this.versicherungsnummer = versicherungsnummer;  
    }  
    public void krankheitenHinzufuegen(Krankheit krankheit){  
        krankheiten.add(krankheit);  
    }  
    public void heilen(Krankheit krankheit){  
        krankheiten.remove(krankheit);  
    }  
    public boolean hatKrankheit(Krankheit krankheit){  
        return krankheiten.contains(krankheit);  
    }  
    public Collection<Krankheit> getKrankheiten(){  
        return krankheiten;  
    }  
}
```

Geeignete Implementierung
wählen, aber nicht öffentlich machen

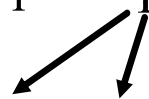
Falls als Implementierung ein TreeSet gewählt wird,
müssen die Elemente das Interface Comparable
implementieren

```
public class Krankheit implements Comparable {  
    private int inkubationszeit;  
    private String name;  
    private String code;  
    private Patient patient;  
    public Krankheit(int inkubationszeit, String name, String code, Patient patient) {  
        this.inkubationszeit = inkubationszeit;  
        this.name = name;  
        this.code = code;  
        this.patient = patient;  
    }  
    public int getInkubationszeit() {  
        return inkubationszeit;  
    }  
    public void setInkubationszeit(int inkubationszeit) {  
        this.inkubationszeit = inkubationszeit;  
    }  
    public String getName() {  
        return name;  
    }  
    public void setName(String name) {  
        this.name = name;  
    }  
    public String getCode() {  
        return code;  
    }  
    public void setCode(String code) {  
        this.code = code;  
    }  
    public int compareTo(Object o) {  
        Krankheit andereKrankheit = (Krankheit)o;  
        //return andereKrankheit.getName().compareTo(name);  
        int andereInkubationszeit = andereKrankheit.getInkubationszeit();  
        if (andereInkubationszeit > inkubationszeit){  
            return -1;  
        } else if (andereInkubationszeit < inkubationszeit){  
            return 1;  
        } else {  
            return 0;  
        }  
    }  
}
```



Abbilden von 1:n-Beziehungen mit Maps Datentyp von Schlüssel und Wert

```
public class Patient {  
    public String name;  
    public int versicherungsnummer;  
    private Map<String,Krankheit> krankheiten = new HashMap<String,Krankheit>();  
  
    public Patient(String name, int versicherungsnummer) {  
        this.name = name;  
        this.versicherungsnummer = versicherungsnummer;  
    }  
  
    public void krankheitenHinzufuegen(Krankheit krankheit){  
        krankheiten.put(krankheit.getCode(),krankheit);  
    }  
  
    public void heilen(String code){  
        krankheiten.remove(code);  
    }  
  
    public void heilen(Krankheit krankheit){  
        krankheiten.remove(krankheit.getCode());  
    }  
  
    public boolean hatKrankheit(String code){  
        return krankheiten.containsKey(code);  
    }  
  
    public Collection<Krankheit> getKrankheiten(){  
        return krankheiten.values();  
    }  
  
    public Collection<String> getCodes(){  
        return krankheiten.keySet();  
    }  
}
```



Schlüssel der Map

Werte der Map

API des Java Collection Frameworks

Collection

- add: boolean
- remove: void
- contains: boolean
- toArray: Object[]
- size: int

List (zusätzlich)

- add(int index, Object o): void
- remove(int index): void
- indexOf(Object o): int

Map

- put(Object key, Object value): void
- contains(Object key): void
- remove(Object key): void
- get(Object key): Object

Queue

	wirft Fehler	"Spezialwert"
Einfügen	add	offer
Entfernen	remove	poll
Existenz prüfen	element	peek