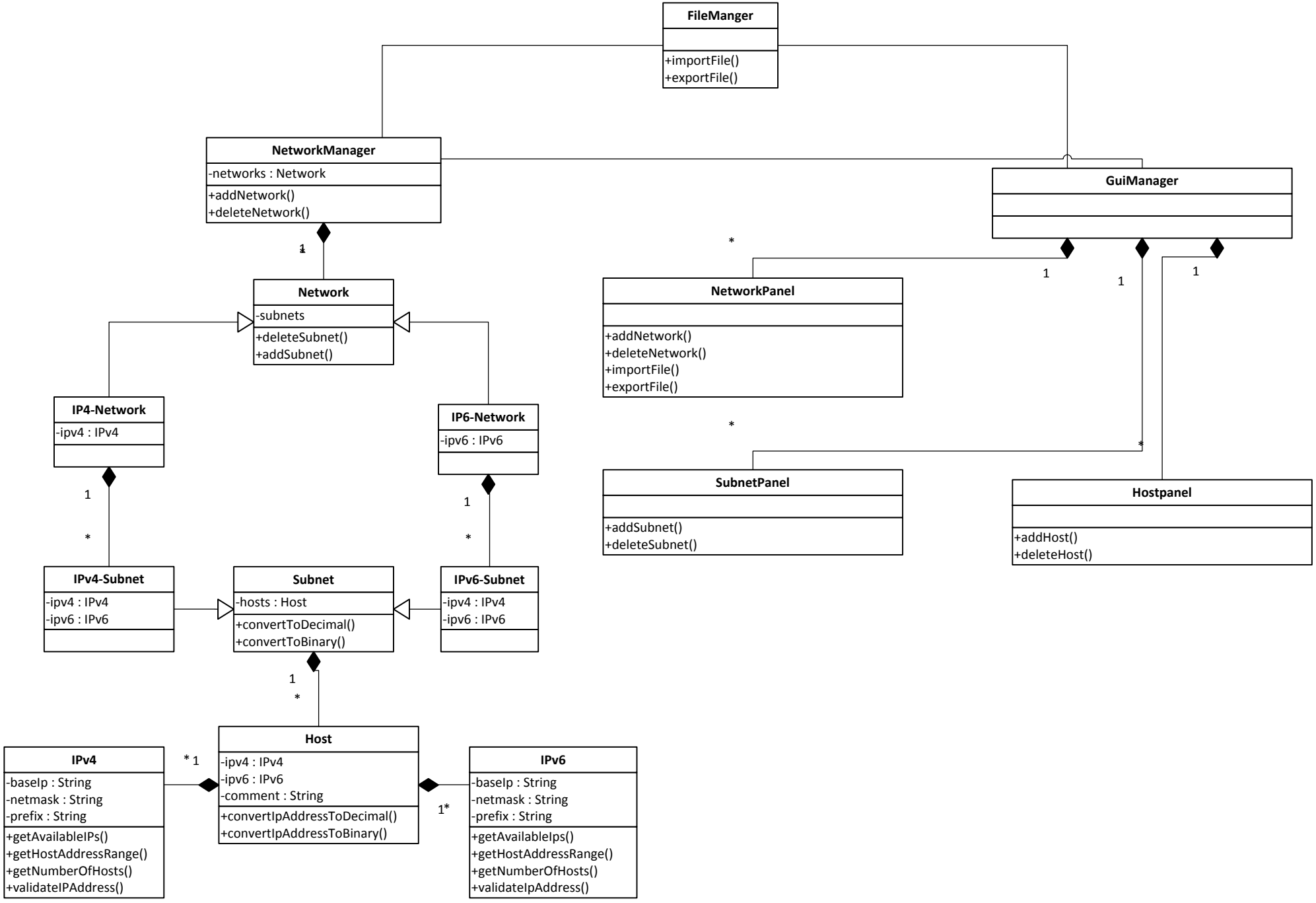
Projekt 04 – Dokumentation zum Klassendiagramm

IT4L - Claas Sündermann – Andrei Wilkens – Garrit Schröder – Carsten Kock

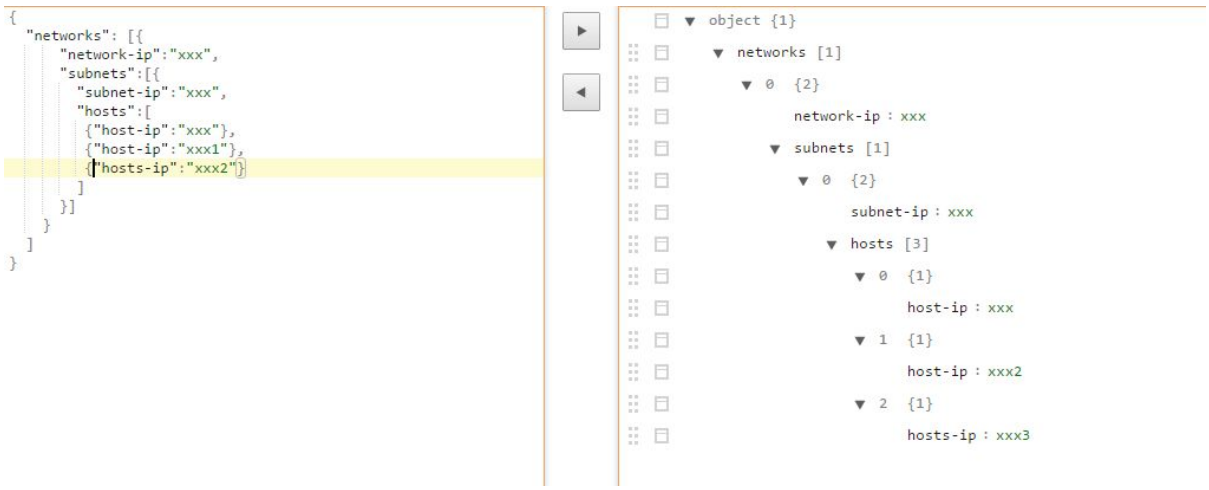
07.06.2016


FileManger
+importFile()
+exportFile()
NetworkManager
-networks : Network
+addNetwork()
+deleteNetwork()
GuiManager
Network
-subnets
+deleteSubnet()
+addSubnet()
NetworkPanel
+addNetwork()
+deleteNetwork()
+importFile()
+exportFile()
IP4-Network
-ipv4 : IPv4
IP6-Network
-ipv6 : IPv6
SubnetPanel
+addSubnet()
+deleteSubnet()
Hostpanel
+addHost()
+deleteHost()
IPv4-Subnet
-ipv4 : IPv4
-ipv6 : IPv6
Subnet
-hosts : Host
+convertToDecimal()
+convertToBinary()
IPv6-Subnet
-ipv4 : IPv4
-ipv6 : IPv6
Host
-ipv4 : IPv4
-ipv6 : IPv6
-comment : String
+convertIpAddressToDecimal()
+convertIpAddressToBinary()
IPv4
-baseIp : String
-netmask : String
-prefix : String
+getAvailableIPs()
+getHostAddressRange()
+getNumberOfHosts()
+validateIPAddress()
IPv6
-baseIp : String
-netmask : String
-prefix : String
+getAvailableIps()
+getHostAddressRange()
+getNumberOfHosts()
+validateIpAddress()

FileManager
Die Klasse FileManager beinhaltet die Funktionen importFile und exportFile.
Die Methode exportFile parsed die Daten aus der Netzwerkliste der Klassse NetworkManager. Die daraus resultierenden Objekte werden mit Hilfe von Jackson 2 in das JSON Dateiformat umgewandelt und abgespeichert.
Die Methode importFile importiert ein gespeichertes Objekt aus dem JSON Dateiformat. Das umgewandelte Objekt wird als Liste von Netzwerken in der Klasse NetworkManager angelegt.

Jackson ist eine Java-Bibliothek welche die Struktur von Objekten und deren Werte in JSON Datenformat umwandelt.

Die im JSON Dateiformat gespeicherten Daten sähen dann z.B. folgendermaßen aus:



Network
In der Klasse network ist eine Liste von Subnets hinterlegt. Dieser können durch die Funktion addSubnet Subnetze hinzugefügt werden außerdem können durch die Funktion deleteSubnet Subnetze aus der Liste entfernt werden. Network steht in einer 1 zu N verbindung mit NetworkManager da 1Netzwerk N Subnetze haben kann. Wobei N den vorgaben des Netzwerkes entspricht.

IPv4 und IPv6
erben die Attribute von Network. Dadurch haben wir die Netze in IPv4 und IPv6 unterteilt.

Subnet
Das Subnet hält folgende Variablen:

1. eine ArrayList<Host> hosts , in welcher die Hosts welche sich in dem Subnet befinden referenziert werden.
2. eine ArrayList<Host> potentialHosts, in welcher die möglichen freien Hosts in dem Subnet referenziert werden.

Das Subnet hat folgende Methoden :

1. public ArrayList<Host> getHosts() - gibt eine Arraylist der Hosts zurück
2. public ArrayList<Host> getPotentialHosts() -gibt eine Arraylist der möglichen Hosts zurück

Ip4_Subnet
Das Ip4_Subnet erbt von Subnet.
zusätzlich besitzt es eine Variable Ipv4 welche vom Typ IPv4 ist.

Ip6_Subnet
Das Ip6_Subnet erbt von Subnet.
zusätzlich besitzt es eine Variable Ipv6 welche vom Typ IPv6 ist.

Host
Ein Host hält 2 Variablen:

1. ipv4 vom typ IPv4
2. ipv6 vom typ IPv6
3. public String Ip4Symbolic - Eine Symbolische Repräsentation der IP4
4. public String Ip4Binary eine Binäre Repräsentation der IP4
5. public String Ip6Symbolic - Eine Symbolische Repräsentation der IP6
6. public String Ip6Binary eine Binäre Repräsentation der IP6

Ein Host hat folgende Methoden:

1. public IPv4 getIpv4() -IPv4 Getter
2. public void setIpv4(IPv4 ipv4) -IPv4 Setter
3. public IPv6 getIpv6() -IPv6 Getter
4. public void setIpv6(IPv6 ipv6) -IPv6 Getter
5. public String getIp4Symbolic() -ip4 symbolic getter
6. public void setIp4Symbolic(String ip4symbolic) -ip4symbolic setter
7. public String getIp4Binary() -ip4 binary setter
8. public void setIp4Binary(String ip4Binary) -ip4 binary setter
9. public String getIp6Symbolic() -ip6 symbolic getter
10. public void setIp6Symbolic(String ip6symbolic) -ip6symbolic setter

11. public String getIp6Binary() -ip6 binary setter
12. public void setIp6Binary(String ip6Binary) -ip6 binary setter

IPv4

Die IPv4 hält folgende Variablen:

1. int baseIPnumeric; - eine numerische Repräsentation der IP
2. int netmaskNumeric; - eine numerische Repräsentation der IP

Die IPv4 hat folgende Methoden:

1. +getAvailableIPs(int) gibt eine anzahl möglicher ips aus
2. +getHostAddressRange() gibt adress range für hosts aus
3. +getNumberOfHosts() gibt aus wieviele mögliche hosts es gibt
4. +validateIPAddress() - Checken ob die Ip adresse Valide ist
5. +getSymbolicIP - gibt die Ip adresse symbolisch aus
6. +getSubnetmasksymbolic -gibt die Subnetzmaske symbolisch aus
7. +getBinaryIp - gibt die Ip Adresse Binär aus
8. +getBinarySubnetmask -gibt die Subnetzmaske Binär aus

IPv6

Die IPv6 hält folgende Variablen:

1. long baseIPnumeric; - eine numerische Repräsentation der IP
2. long netmaskNumeric; - eine numerische Repräsentation der IP

Die IPv6 hat folgende Methoden:

1. +getAvailableIPs(int) gibt eine anzahl möglicher ips aus
2. +getHostAddressRange() gibt adress range für hosts aus
3. +getNumberOfHosts() gibt aus wieviele mögliche hosts es gibt
4. +validateIPAddress() - Checken ob die Ip adresse Valide ist
5. +getSymbolicIP - gibt die Ip adresse symbolisch aus
6. +getSubnetmasksymbolic -gibt die Subnetzmaske symbolisch aus
7. +getBinaryIp - gibt die Ip Adresse Binär aus
8. +getBinarySubnetmask -gibt die Subnetzmaske Binär aus

GUI Manager
Der GUI -Manager registriert alle weiteren GUI -Panels und deren Komponenten. Die Hauptaufgabe des Managers ist eine obere Instanz der GUI zu sein. Alle Interaktionen die auf der GUI getätigt werden, werden im Manager registriert und an die betreffenden Komponenten weitergeleitet.

Network Panel
Das Network Panel besitzt folgende Funktionen:

- Netzwerk hinzufügen
- Netzwerk löschen
- Import einer Datei
- Export in eine Datei

Nach unserem Aufbau sind die GUI -und die Businesskomponenten getrennt. So wird zum Beispiel bei einem Klick auf „Netzwerk hinzufügen“ ein Dialogfeld geöffnet, um ein Netzwerk entgegenzunehmen. Der eingegebene Wert kommt beim Network Panel an und wird an den GUI Manager weitergeleitet. Der GUI Manager wiederum sorgt dafür, dass die Daten an die richtigen Komponenten gegeben werden. In diesem Fall an den Network Manager. Jegliche darauf folgende Aufgaben werden vom Network Manager übernommen. Sollte bei einer der Aufgaben ein Fehler auftreten, wird dieser zurück an das Network Panel gegeben und angezeigt.
Wie an diesem Beispiel gut zu erkennen ist, haben wir bei unserer Planung dafür gesorgt, dass wir eine strikte Trennung zwischen GUI und Model Schicht haben.

Subnet Panel
Das Subnet Panel besitzt folgende Funktionen:

- Subnetz hinzufügen
- Subnetz löschen

Wie beim Network Panel werden auch beim Subnet Panel die Benutzeraktionen registriert und, über den Gui Manager, an die betreffenden Komponenten weitergeleitet.
Das Subnet Panel liefert dem Benutzer eine struckturierte Übersicht aller Subnetze. Die Übersicht bietet dem Benutzer die oben benannten Funktionen.

Host Panel
Das Host Panel besitzt folgende Funktionen:

- Host hinzufügen
- Host löschen