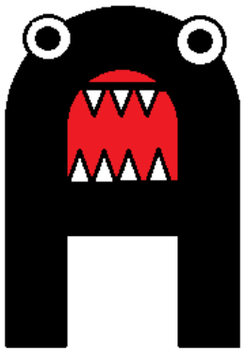
RAGE

Abschlusspräsentation

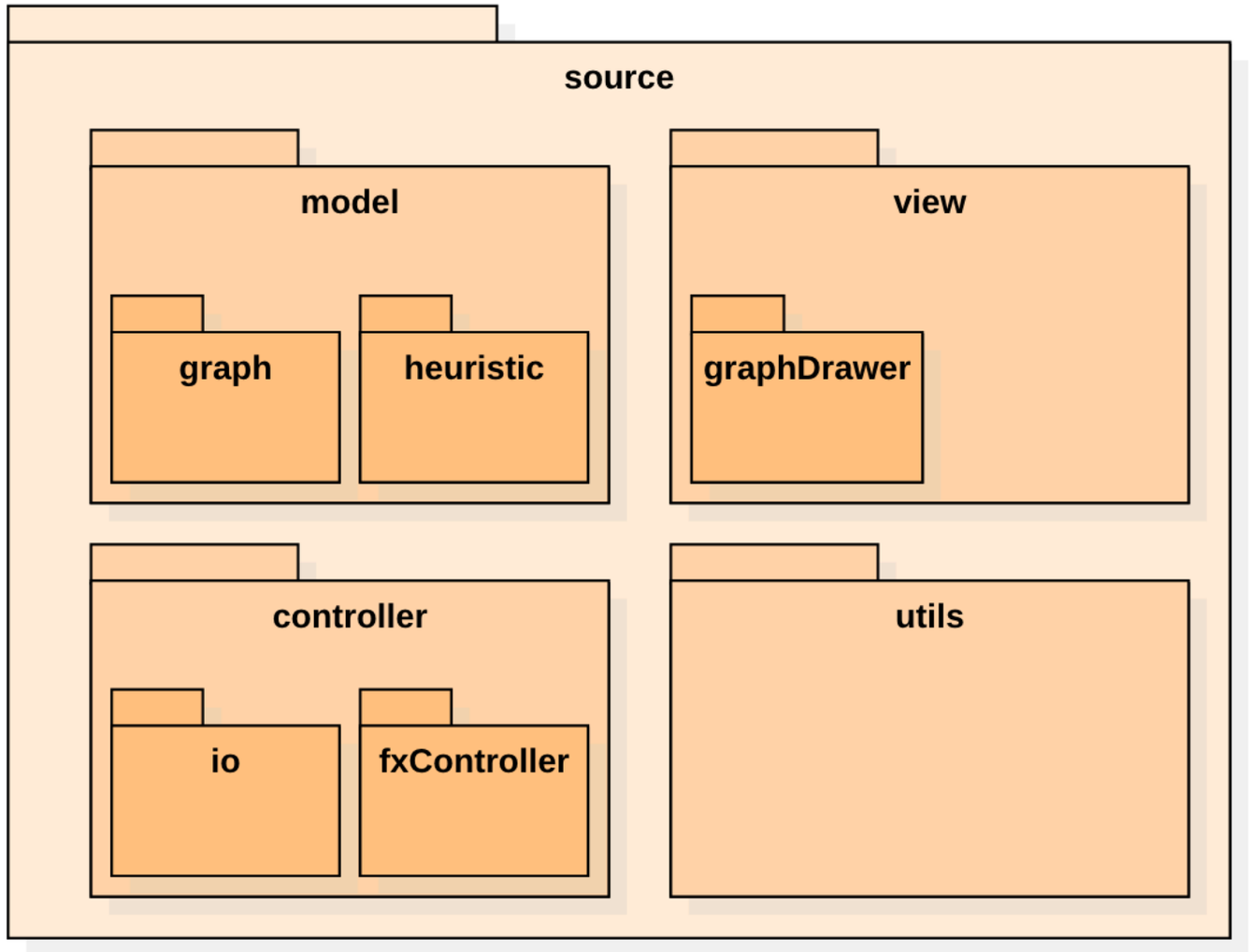
Entwurf

gehalten von Jonas Linßen

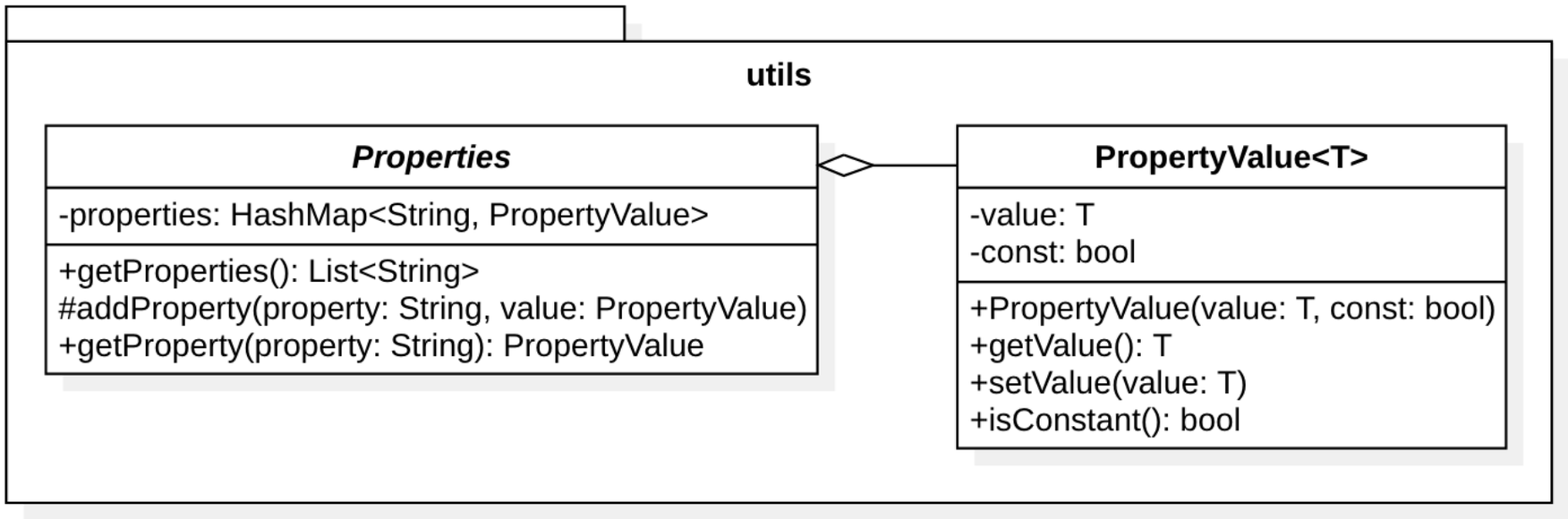
Inhalt



Allgemeiner Entwurf — Paketstruktur



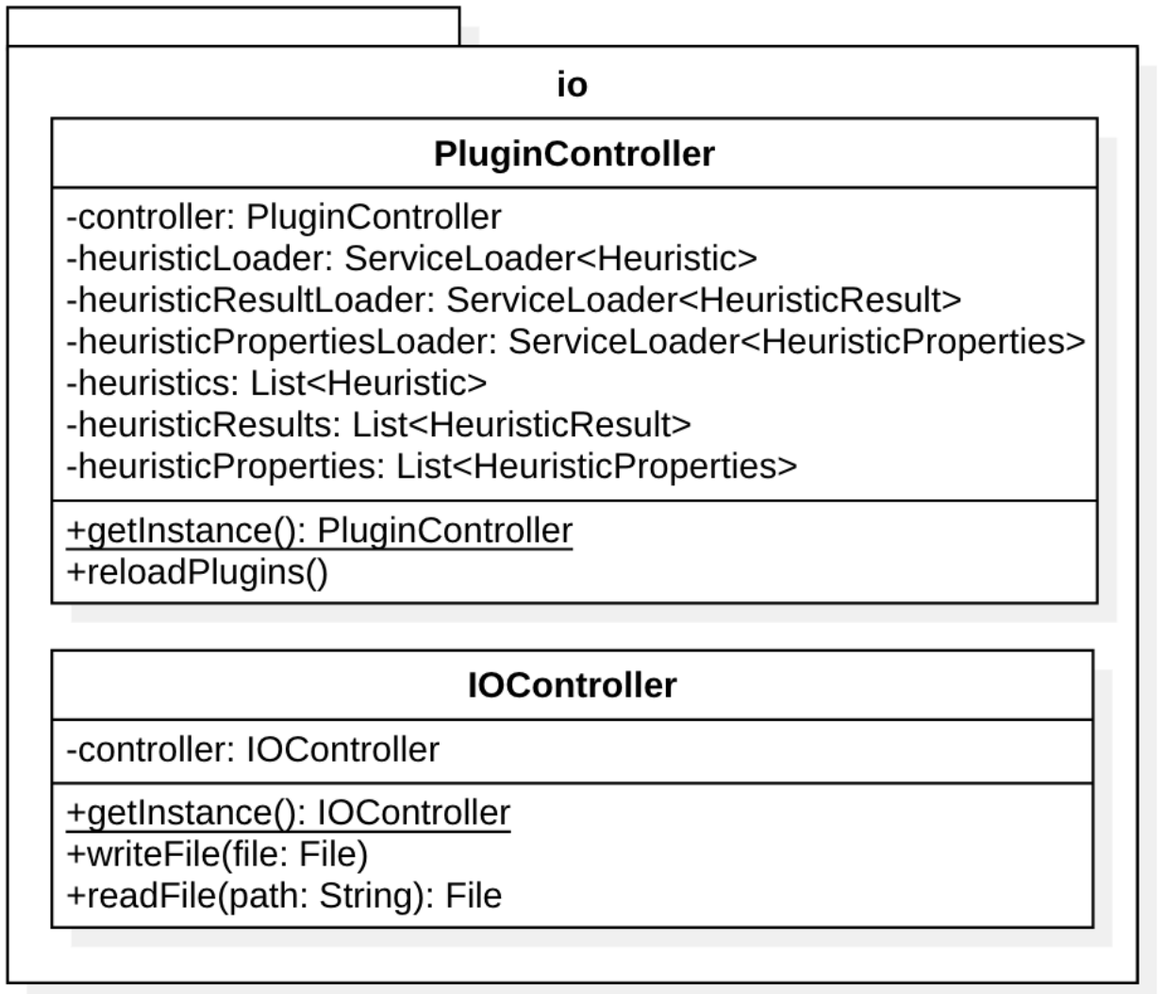
Allgemeiner Entwurf — Properties Klasse



Adaptiver Key-Value Speicher ermöglicht
⟶ Kommunikation über alle Ebenen
⟶ einfache Anpassung der Benutzeroberfläche

Fest spezifizierte Keywords
⟶ ermöglichen einfache Übersetzung der Benutzeroberfläche

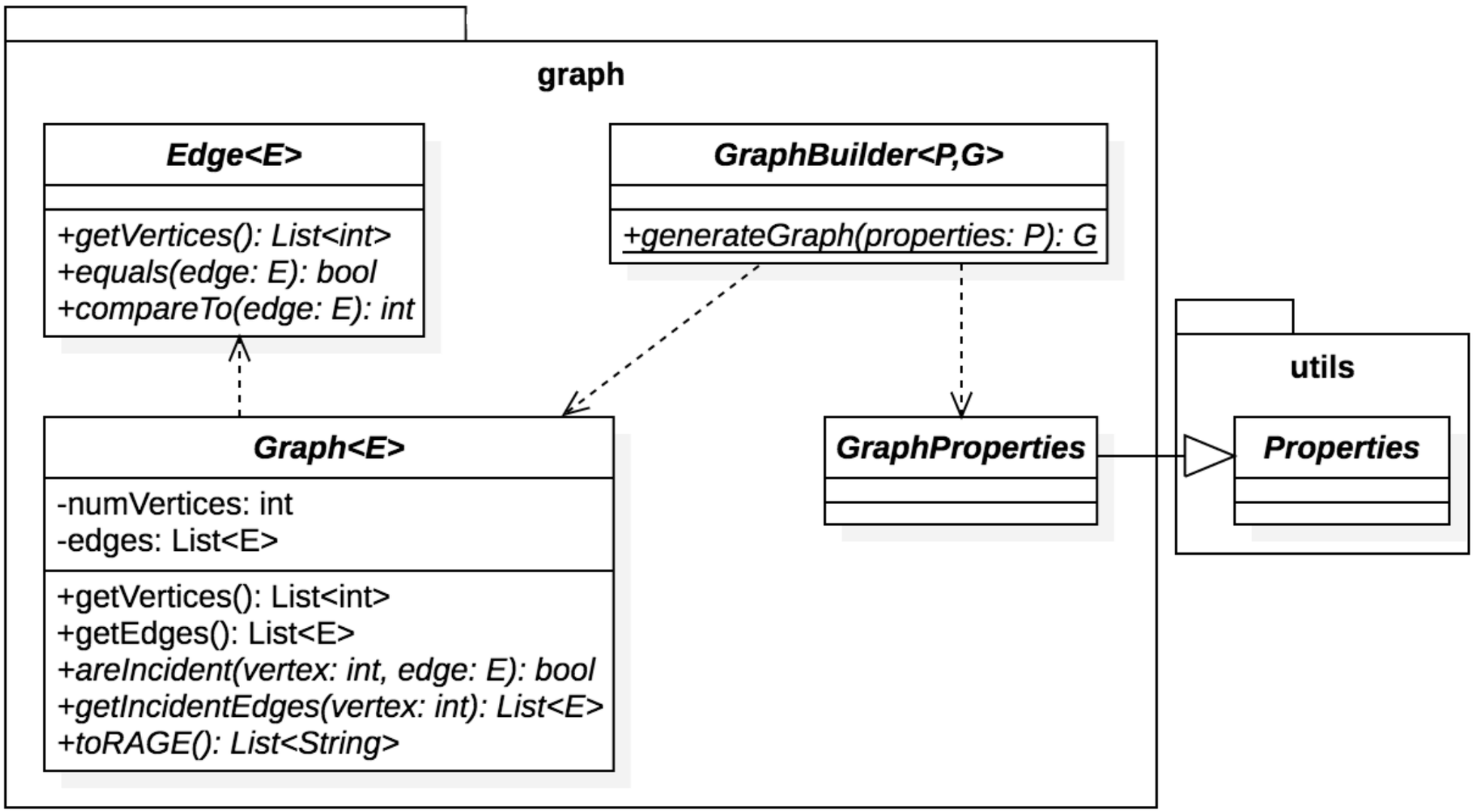
Allgemeiner Entwurf — Speichern und Laden



Singleton - Entwurfsmuster
⟶ ermöglicht Parallelisierung

PluginController behält Übersicht
über alle geladenen Plugins

Model — Graphen



Kapselt die abstrakte Struktur eines Graphen

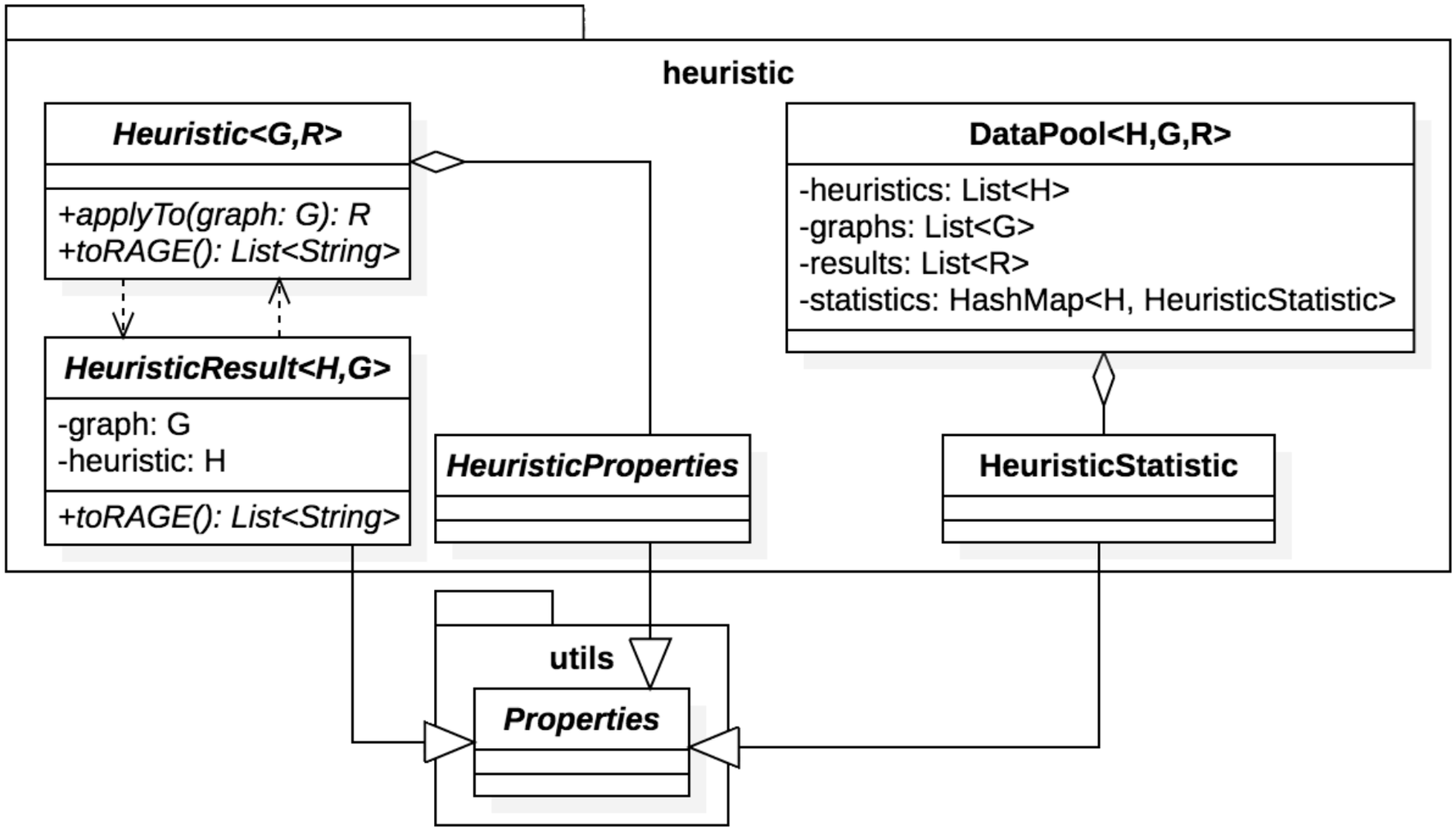
⟶ konkreter Graphentyp über Kantentyp gegeben

⟶ weitere Eigenschaften / Inzidenzrelationen abhängig vom Graph

⟶ Knotenreihenfolge implizit, Kantenreihenfolge explizit

Factory - Entwurfsmuster zur Graphgenerierung

Model — Heuristiken



Heuristik stets abhängig von Graphentyp G und Ergebnistyp R
⟶ Anwendung sollte deterministisch sein

DataPool
⟶ wendet alle Heuristiken auf alle Graphen an
⟶ sammelt Statistiken über die Heuristiken

Model — Heuristiken — EFLGreedyOne

Vermutung:
Jeder einfache Hypergraph mit n Knoten hat eine valide Kantenfärbung mit n Farben

temporäres Speichern der Hyperkanten mit genau einer freien Farbe

```
For any vertex v in order of a breadth first search
   for hyperedge e incident to v
      while there are hyperedges f with exactly one free color
         take that f with minimal index
         color f with the minimally used free color of f
      if e is colored
         continue
      if e cannot be colored
         return incomplete coloring
      color e with the minimally used free color of e
return complete coloring
```

Model — Heuristiken — TCMixedGreedySet

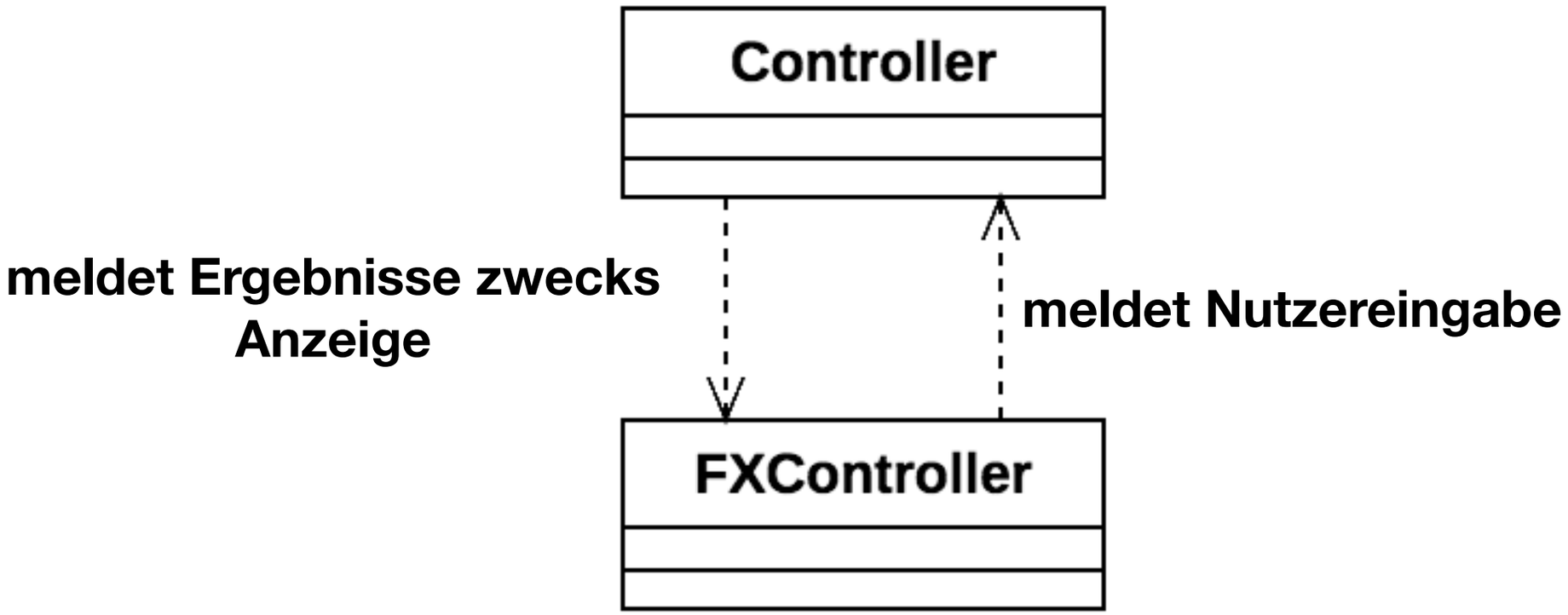
Vermutung:
Jeder einfache ungerichtete Graph mit Maximalgrad D hat eine valide Totalfärbung mit D+2 Farben

temporäres Speichern von Teilmengen der zu einem gemeinsamen Knoten inzidenten und unkolorierten Kanten mit minimaler Flexibilität
falls Knoten ungefärbt: Varianten mit und ohne Knoten

```
while there is a vertex with such a minimal set
   find set X of minimal flexibility belonging to a vertex v
   with minimal index
   if it has negative flexibility
      return incomplete coloring
   if v is uncolored
      if v cannot be colored
         return incomplete coloring
      color v with minimally used free color
   for edge e in X
      if e cannot be colored
         return incomplete coloring
      color e with minimally used free color
return complete coloring
```

Controller / View — Beschreibung

Kontrolliert Datenverarbeitung



Zuständig für Nutzerinteraktion + Darstellung

Controller / View — TabController

TODO

Controller / View — Graph-Editor

TODO

Nutzerinteraktion über VisualGraph
⟶ nach Bestätigung Umwandlung zu zugehörigem Graphentyp des Models

Sequenzdiagramme — Graphgenerierung

TODO