HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

ÜBUNGSSERIE 2

Algorithmen & Datenstrukturen AD2 / HS 2018

AD2 Team

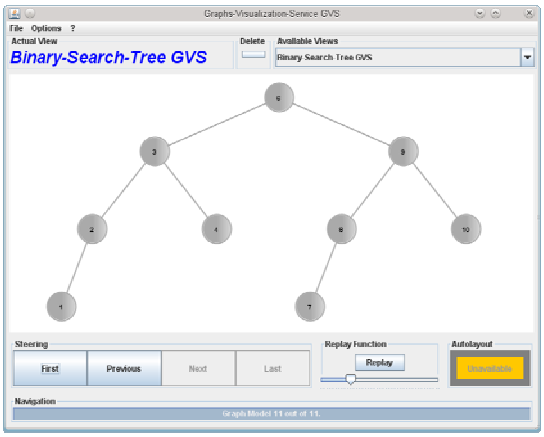
Aufgabe 1

Ein zufällig erzeugter binärer Suchbaum mit n Knoten hat eine Höhe von ca. *C*log n*.
Sie sollen diese Aussage experimentell bestätigen.

Erzeugen Sie grosse binäre Suchbäume mit n Elementen und bestimmen Sie deren Höhe.
Um die Suchbäume aufzubauen, erzeugen Sie Zufallszahlen zwischen 0 und MAXINT und bestimmen Sie die Konstante *C* näherungsweise.
Dazu müssen Sie eine Klasse für binäre Suchbäume mit geeigneten Methoden implementieren.
Implementieren Sie dazu die Klasse *BinarySearchTree* (siehe Skripte-Server).
Hinweis: Zusätzliche Methoden (interne) können resp. sollen hinzugefügt werden.

Optional:
Visualisieren Sie den Baum mit dem *Graphs-Visualization-Service GVS.*

a) Auspacken: *7_Zusatzmaterial/GVS/GraphsVisualizationService-GVS_v1.6.zip*
   Hinweis: Der Pfad wo GVS liegt darf keine Umlaute und/oder Leerschläge enthalten!
b) Starten des Visualierungs-Server *GVS_Server_v1.4.jar* (Doppel-Klick auf die Datei oder in der Konsole mit `java –jar GVS_Server_v1.4.jar`)
c) Die drei *jar*-Files *GVS_Client_Socket_v1.6.jar, log4j-1.2.12.jar* und *dom4j-1.6.1.jar* im Verzeichnis *GVS/lib* in das aktuelle Eclipse-Projekt einbinden
   (im Standard-AD2-Eclipse-Projekt bereits gemacht):
   Context-Menü des aktuellen Projektes (rechte Maus) > Build Path > Configure Build Path... : Libararies : Add External JARs …
d) Das Eclipse-Projekt im Verzeichins *GVS* importiert werden mit:
   Menü:File>Import...>General>Existing Projects into Workspace
e) In jenem Eclipse-Projekt *GvsExample* das Programm BaumTest.java ausführen lassen:



Advanced:
Realisieren Sie die Anbindung an den *Graphs-Visualization-Service* ohne irgendeine Abhängigkeit zu GVS innerhalb von *BinarySearchTree.java* (die Klasse *BinarySearchTree* weiss nichts von GVS, aber der Baum wird dennoch visualisiert! ;-)