|  |
| --- |
| Högskolan i Gävle |
| Projektrapport |
| Datavisualization - Design and Construction, 7,5 hp  HT 19, period 1  Visualisering av källkod  av  *Thomas Lundgren*  *900802-1835* [*thomaslundgren@live.com*](mailto:thomaslundgren@live.com)  *Niklas Nordgren*  *901010-5758* [*n\_nordgren@hotmail.com*](mailto:thomaslundgren@live.com)  *Hanna Medén*  *960214-0965* [*hannameden@hotmail.com*](mailto:hannameden@hotmail.com) |
|  |
| **Institutionen för matematik, natur- och datavetenskap**  **Högskolan i Gävle**  S-801 76 Gävle, Sweden |
|  |
|  |

**Innehållsförteckning**

[1 Inledning 1](#_Toc21691425)

[1.1 Syfte och riktlinjer 1](#_Toc21691426)

[1.2 Användningsfall och specifikation av grafiskt användargränssnitt 1](#_Toc21691427)

[2 Metod och utförande 3](#_Toc21691428)

[2.1 Arbetsflöde 3](#_Toc21691429)

[2.2 Implementation 4](#_Toc21691430)

[2.2.1 Gränssnitten PropertyChangeBroadcaster och PropertyChangeListener 4](#_Toc21691431)

[2.2.2 Domänmodellen 4](#_Toc21691432)

[2.2.3 Klassen ClockController 5](#_Toc21691433)

[2.2.4 Användargränssnittets stomme – klassen Gui 5](#_Toc21691434)

[2.2.5 Grafisk representation av digital och analog klocka 5](#_Toc21691435)

[2.2.6 Klassen ControlPanel 6](#_Toc21691436)

[2.2.7 Klassen AlarmPanel 6](#_Toc21691437)

[2.2.8 Klassen AlarmEntryPanel 7](#_Toc21691438)

[2.2.9 Interfacet GuiAlarmAction och klasserna GuiSoundAlarmAction och GuiBlinkAlarmAction 8](#_Toc21691439)

[2.2.10 Klassen FileManager 8](#_Toc21691440)

[3 Resultat 9](#_Toc21691441)

[3.1 Användning av designmönstren Model-View-Controller (MVC) och Observer 9](#_Toc21691442)

[3.2 Användning av javax.swing- och java.awt-klasser för att skapa ett grafiskt användargränssnitt 9](#_Toc21691443)

[3.3 Implementation av användningsfall 10](#_Toc21691444)

[3.4 Körning av programmet AlarmClock App 11](#_Toc21691445)

[4 Diskussion 14](#_Toc21691446)

1. Inledning
   1. Bakgrund

Källkod bör vara enkel att upprätthålla för att minimera tidsåtgången av mjukvaruutveckling. Faktorerna som påverkar detta är bland annat kodbasens klassrelationer, komplexitet och struktur.

Visualiseringar av källkod ger utvecklare och andra intressenter en bättre överblick om källkodens skick, samt om det förekommer brister som måste åtgärdas.

* 1. Syfte

Syftet med detta projekt var att undersöka möjligheterna att visualisera källkod i dataanalysverktyget Matlab.

* 1. Frågeställning

För att avgränsa arbetet valdes tre olika aspekter av en kodbas som skulle visualiseras:

* Relationer mellan klasser i en Java-kodbas.
* Bidragen till ett Git-repository över tid.
* Storleken hos en kodbas.

Avgränsningen resulterade i följande frågeställningar:

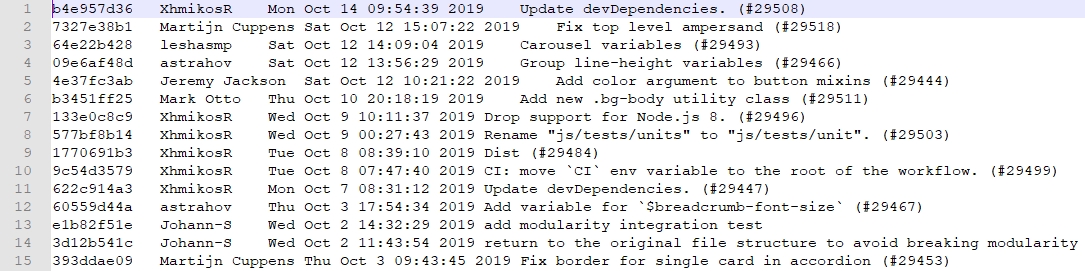
* Hur visualiseras relationer mellan klasser i Java-källkod med användning av en riktad graf på bästa sätt?
* Hur visualiseras relationer mellan klasser i Java-källkod med användning av en anropsmatris (eng. adjacency matrix) på bästa sätt?
* Hur visualiseras bidragen till en kodbas över tid med hjälp av bidragshistorik framtaget med versionshanteringssystemet Git på bästa sätt?
* Hur visualiseras storleken hos en kodbas och hur jämförs den med andra kodbasers storlek på bästa sätt?

1. Metod
   1. Visualisering av bidragshistorik till Git-repository

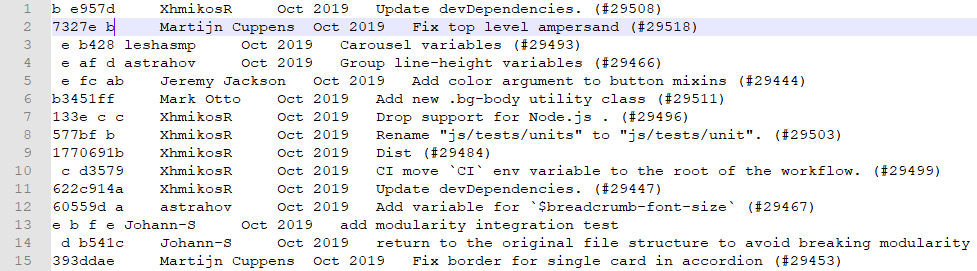
Ett program som visualiserar bidragshistorik till ett Git-repository skrevs i Matlab.

* + 1. Importera genererade data till Matlab

Git lagrar all bidragshistorik förknippat med ett repository i loggfiler. För att generera data över bidragshistoriken användes kommandot git log i Git Bash. Det repository som användes var CSS-ramverket Bootstrap. Resultatet skrevs till en textfil med tabbseparerade värden (TSV-fil). Figur 1 visar innehållet i textfilen.

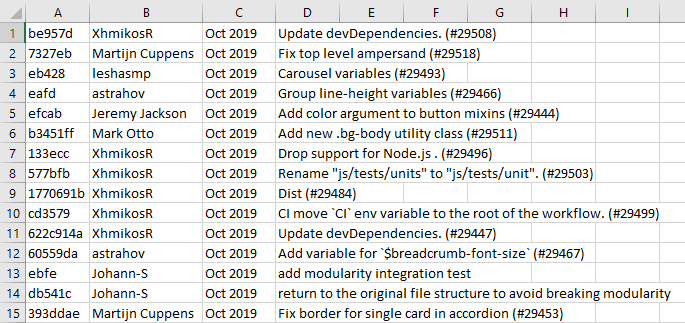
Innehållet i textfilen manipulerades genom Notepad++:s funktion ”hitta och ersätt”. Ett reguljärt uttryck användes som ersatte veckodag, datum, klockslag och kolontecken med blanksteg i kolumnen som innehåller datum.

Figur : Resultatet av att köra kommandot git log i Git bash.



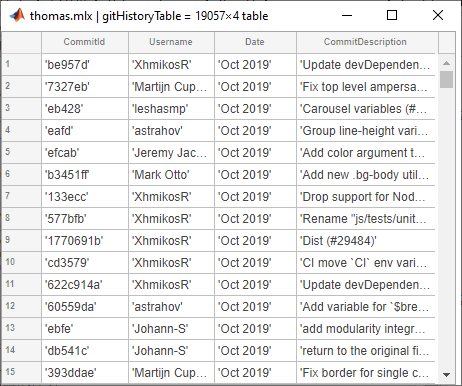
Figur : TSV-filen efter att funktionen ”hitta och ersätt” har använts.

TSV-filen öppnades sedan i Microsoft Excel och sparades i xlsx-format (Exceldokument-format), se Figur 3.



Innehållet i Excel-filen importerades slutligen till Matlab genom funktionen readtable(). Resultatet redovisas i Figur 4.

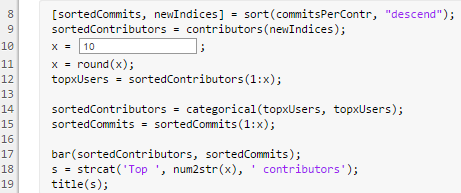
Figur : Innehållet i TSV-filen har importerats till Microsoft Excel.



Figur : Innehållet i Excel-filen importerad som en tabell i Matlab.

* + 1. Diagram över de bidragsgivare som bidragit mest

Koden för att skapa ett diagram över de bidragsgivare som gjort flest bidrag till Bootstrap:s repository redovisas i Figur 5.

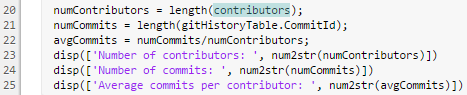


Figur : Koden för att skapa ett stapeldiagram över de användare som bidragit mest.

Koden grupperar antalet bidrag (eng. commits) efter bidragsgivare med funktionen histcounts(), sorterar efter antalet bidrag i fallande ordning och skapar ett stapeldiagram med funktionen bar().

* + 1. Utskrift av antalet bidragsgivare, antalet bidrag och bidrag per bidragsgivare

Programmet skriver ut antalet bidragsgivare, antalet bidrag och bidrag per bidragsgivare (Figur 6).



Figur : Koden för att skriva ut antalet bidragsgivare, antalet bidrag och bidrag per bidragsgivare.

* + 1. Visualisering av antalet bidrag över tid

För att visualisera antalet bidrag till Bootstrap:s kodbas över tid

