|  |
| --- |
|  |
| Evil Corporation Calculator |
|  |
| Grupparbete |

**SYSTEMINTEGRATÖR YHSIPI17/ARBETSMETODIK FÖR UTVECKLARE**

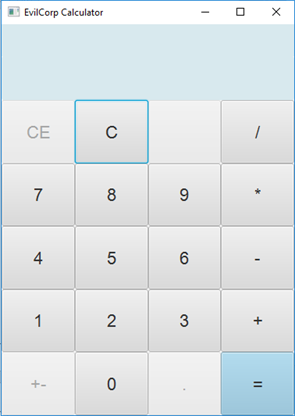
den 5 februari 2018

Billy Andersson, Tommy Eriksson, Malin Albinsson

Evil Corporation Calculator

Grupparbete

# Sammanfattning

Sammanfatta kort det ni gjort, gör er läsare intresserad

Detta är en enkel miniräknare byggd i Javafx som klarar att hantera de fyra räknesätten för heltal.

TODO

# Inledning/Syfte

Presentera frågeställningar eller mål som besvaras i arbetet

Vår uppgift var att leverera källkod och ett färdigbyggt applikationspaket till uppdragsgivaren Evil Corporation som behövde en miniräknare som minst klarade av de fyra räknesätten. Källkoden skulle versionshanteras med git och vi behövde skapa ett repository för detta på github.com.

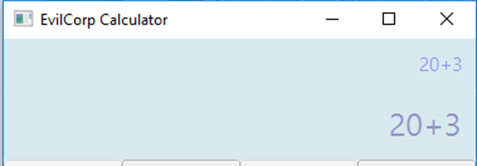
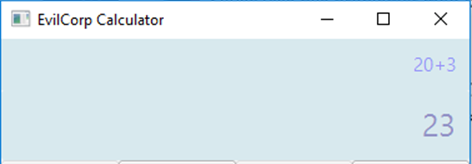
TODO

Genomförande  
Här hamnar det mesta kring ert genomförande, det ni arbetat fram helt enkelt. Presentera tabeller,illustrationer här.

//TODO Uppgiften planerade vi, delade in arbetet och genomförde såhär…

Vi valde en MIT-licens därför att detta är en enkel applikation där källkoden kommer vara åtkomlig för alla via github. Det är inte heller något nytt, komplicerat eller innovativt så det finns anledning till större begränsningar av den anledningen. //TODO …

//TODO Kalkylatorns funktion (bilder pilar förklaringar)…

TODO Djupare förklaring av kod om nödvändigt…

# Avgränsning

Vilka features har ni valt att inte implementera? Hur avgränsade ni ert arbete?

Detta är en enkel kalkylator som vid första leverans klarar att hantera de fyra räknesätten för heltal, men däremot inte kan hantera decimaltal, parenteser eller operatorprioritet. Negativa tal hanteras inte heller. CE-knappen (clear entry) är inte implementerad. Knappar för icke-implementerad funktionalitet har ljusgrå text. //TODO ändra om något av dessa implementeras innan release.

TODO Fortsätt här

# Slutsatser/diskussion

Ta upp fördelar och nackdelar med er implementation. Diskutera era nya insikter om själva projektet och inte era personliga förkunskaper.

TODO Fördelar med vår implementering är… En fördel med att arbeta med github är att det är lätt att ta del av vad de andra har gjort…

TODO En nackdel med vår implementering är att det kan uppfattas förvirrande när ekvationen visar t ex 6+4\*2 och resultatet blir 20 (eftersom operatorprioritet inte hanteras) då man är van att multiplikation går före addition.

En nackdel med vårt upplägg var att vi inte alltid var tillräckligt tydliga med vem som skulle jobba med vad, så de blev en del dubbelarbete.

En viktig insikt är att man vid ett samarbetsprojekt där man sitter på olika ställen och dessutom arbetar på olika tider vinner mycket på att redan innan man börjar tydliggöra exakt hur den process man tänkt jobba efter ska se ut. Vi hade varsin utvecklings-branch och en gemensam testing-branch (och förstås gemensam master), men vi kunde varit tydligare i vem som skulle ta ansvar för vad, hur vi skulle ta oss mellan brancherna och hur besluten skulle dokumenteras. Issue-hanteringen hade vi också kunnat definiera tydligare. Två orsaker till att vi inte lyckades få till en mer optimal arbetsprocess var att både git och github är relativt nya för oss, och att vi inte tog oss tillräckligt med tid i början för att sätta upp en genomtänkt process. Vi var mer fokuserade på funktionalitet i applikationen.

Om vi skulle göra om projektet idag skulle vi utse en projektledare som fick ha mer fokus på processen medan de andra ägnade sig åt själva koden.

TODO Slutresultatet blev... som förväntat/sämre/bättre.

# Referenser

TODO