jKalc

Avancerad miniräknare

Jonas Krook
861212-4837

Introduktion

De flesta operativsystem har någon form av räknare tillgänglig för användaren. Dessa är vanligtvis lika till funktion, där det som varierar är vilka matematiska funktioner som är tillgängliga och huruvida användaren kan använda parenteser. Något som de flesta saknar är möjligheten att spara värden i variabler, och när väl möjligheten finns så är det oftast begränsat till ett fåtal variabler med förbestämda namn.

En användare som frekvent beräknar en serie operationer har oftast ingen som helst möjlighet att spara denna serie, utan får istället utföra räkneoperationerna om igen för varje beräkning. Möjligheten för användaren att själv spara och definiera funktioner och beräkningsserier saknas oftast.

För att få möjlighet att använda variabler, funktioner och skript-filer finns alternativen att använda programmeringsspråk eller beränkingsprogram som Matlab, Mathematica eller GNU Octave. För en användare som behöver en litet kraftfullare miniräknare har troligtvis inte mycket att hämta i ett renodlat programeringsspråk. Dels blir användaren tvungen att lära sig ett programmeringsspråk för att kunna utföra rent matematiska beräkningar. Oftast skall mycket deklareras och initieras i ett programmeringsspråk, vilket medför en onödig uppstartstid innan en beräkning kan utföras.

Många beräkningsprogram försvinner för hemmaanvändare på grund av priset, men även om GNU Octave är fri programvara så är programmet långt mer avancerat än vad som fordras i många matematiska beräkningar som hemmaanvändare kan tänkas vilja göra. Dessutom är beräkningsprogram långt mer omfattande och minneskrävande än vad som är erforderligt.

Det finns alltså en lucka mellan miniräknare och beräkningsprogram. En räknare som har möjlighet att använda variabler, som kan läsa filer med flera beräkningsoperationer och som inte kräver lika mycket systemresurser som moderna beräkningsprogam kan fylla en lucka och vara användbar för avancerade hemmaanvändare.

Klasserna som inte har med GUI att göra går med fördel att porta till andra plattformar, som exempelvis mobila eller webbaserade applikationer.

Mål

Målet med jKalc är att skapa en avancerad miniräknare som fungerar som en blandning av beräkningsprogram och miniräknare. Räknaren skall kunna läsa en textsträng, avgöra om den är ett tillåtet uttryck och beräkna värdet av uttrycket. Användargränssnittet skall visa användaren vilka variabler som är deklarerade för tillfället och vad de har för värde, en historia över senaste operationer samt en lista med sparade skriptfiler.

Krav

Antalet inbyggda operatorer och funktioner som är tillåtna i ett uttryck kommer att bestämmas av den tillgängliga tiden för implementering, men absolut minst så skall ett tillåtet uttryck kunna bestå av addition, subtraktion, multiplikation, division, tilldelning och sinusfunktionen. Skriptfiler

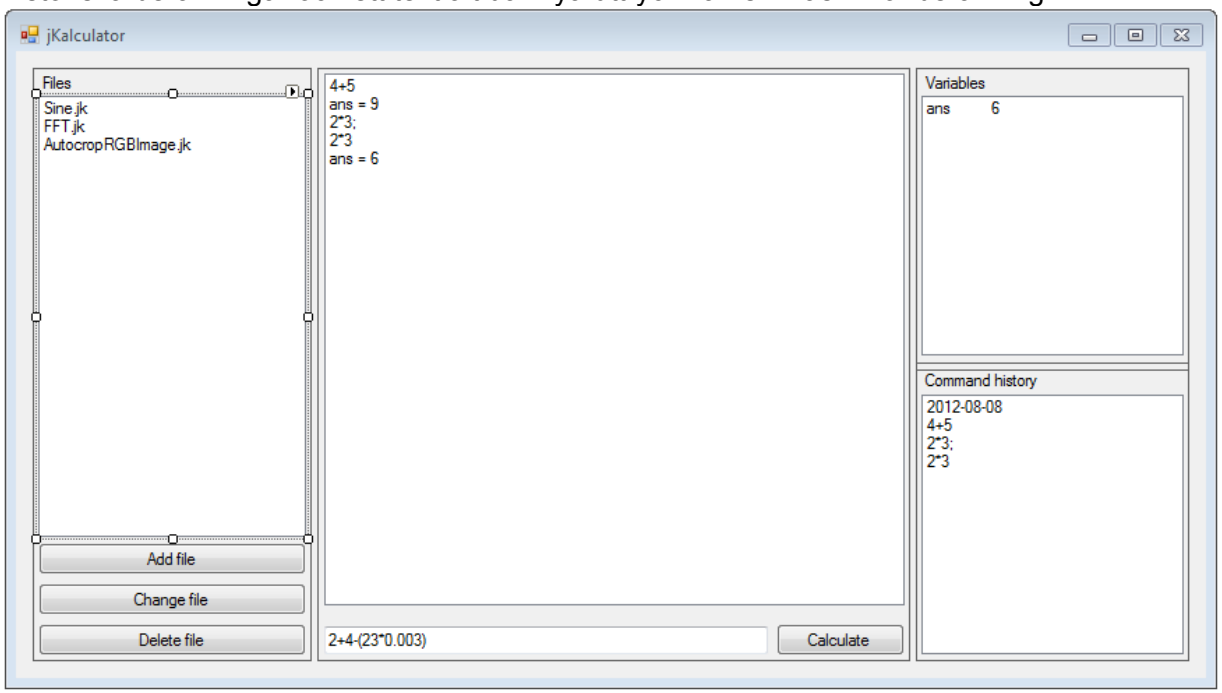
skall bestå av tillåtna uttryck som är separerade med radbrytningar. Huruvida ett uttryck är tillåtet avgörs av matematiska räkneregler.

Felaktiga uttryck skall hanteras utan att programmet kraschar. Beräkningsmässiga fel, som till exempel division med noll, skall hanteras av programmet utan att det kraschar. Ett felmeddelande skall visas för användaren.

Klassdiagram

Användargränssnitt

Räknarens huvudklass är en Form som utgör huvudfönstret varifrån alla relevanta objekt skapas, händelser hanteras och funktioner anropas. Huvudfönstret innehåller en lista på skriptfiler, en lista på tidigare beräkningsresultat, en lista på deklarerade variabler, en lista på historiska beräkningar och ett textfält där nya uttryck kan skrivas in för beräkning.



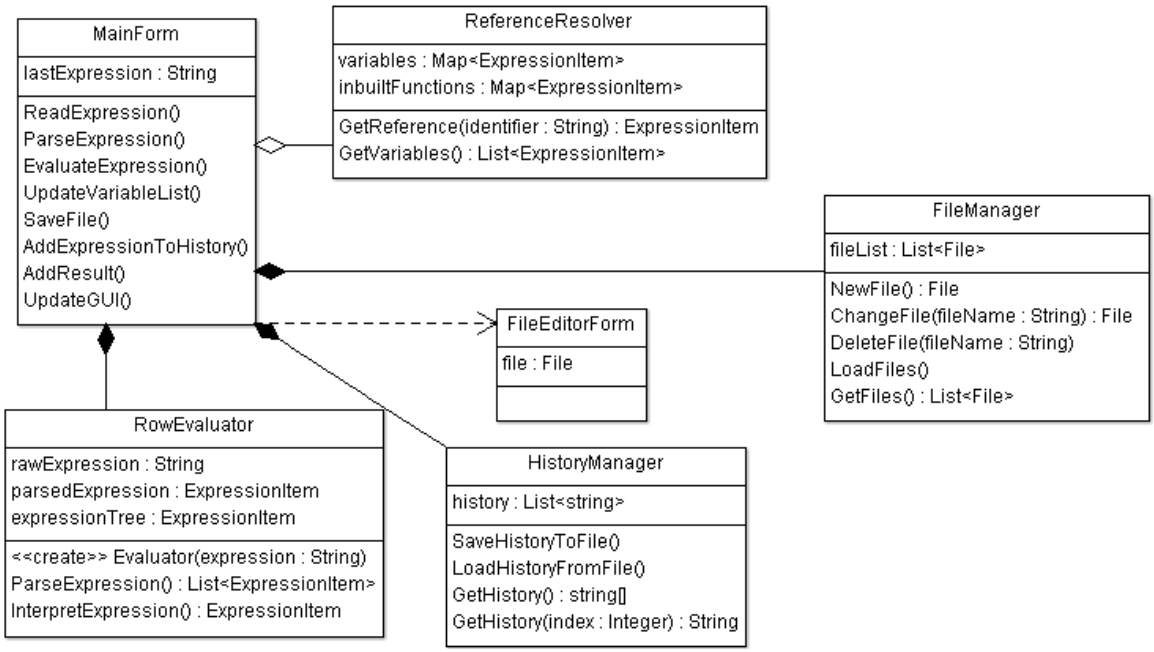
Hantering av filer sker av ett objekt av klassen FileManager. FileManager läser in alla filnamn, öppnar filer, ändrar filer och tar bort filer. För att ändra filer så öppnas genom huvudfönstret ett till fönster där möjlighet för redigering finns.

Alla variabler hanteras av ett objekt av klassen ReferenceManager. Det objektet hanterar alla referenser och innehåller objekt för variabler och funktioner. Huvudfönstret använder det objektet för att få en lista på alla deklarerade variabler och deras värden.

Historiska beräkningar hanteras av ett objekt av klassen HistoryManger. Historien består av allt som skrivs i textfältet för uttryck, och sparas i en textfil mellan körningar. Objektet som hanterar historiska beräkningar sköter inläsning av historian från fil, nytillkomna beräkningar och sparning till fil.

Listan över beräkningsresultat hanteras av huvudfönstret. När ett uttryck matas in så kopieras texten och det tillhörande beräkningsresultatet till listan över beräkningsresultat. Beräkningsresultaten sparas inte mellan körningar.

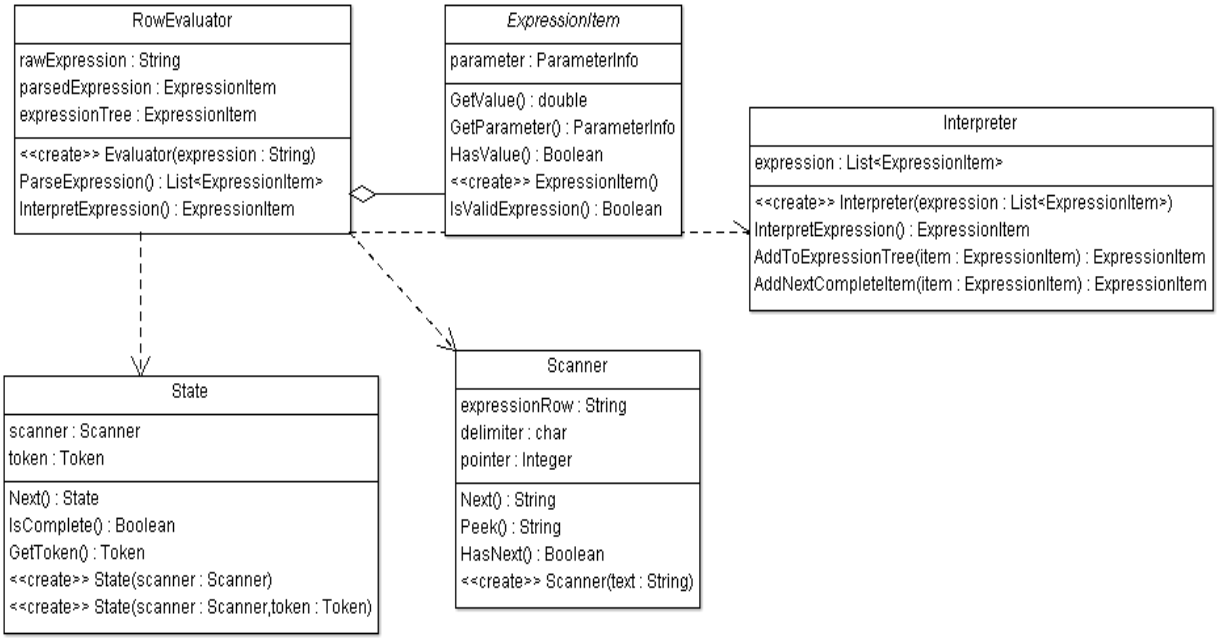
När ett nytt uttryck matas in i textrutan så skapas ett objekt av klassen RowEvaluator. Det objektet är ansvarigt för att tolka uttrycket, bedöma om det är korrekt och beräkna resultatet. Huvudfönstret ser sedan till att visa resultatet. Huvudfönstrets klass och relationerna till de andra klasserna visas i figur 1.



Figur 1. GUI

Hantering av uttryck

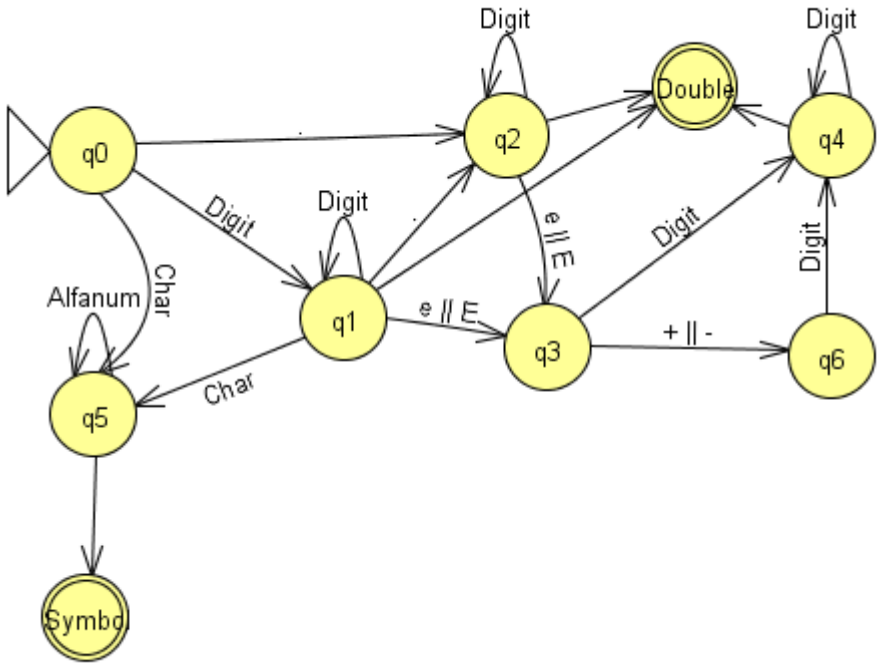
När ett uttryck matas in i huvudfönstret så skickas texten till en RowEvaluator. RowEvaluator beräknar ett textuttryck i tre steg. Först skannas alla tillåtna symboler i textuttrycket. Här skapas ett objekt för varje identifierad del av textuttrycket och placeras i en lista. Listan med symboler görs sedan om till en lista med uttrycksdelar, ExpressionItem. Listan med ExpressionItem skickas sedan till ett objekt som tolkar uttrycket och beräknar det om det är ett tillåtet uttryck. I figur 2 kan relationerna för RowEvaluator ses.



Figur2. ExpressionEvaluator

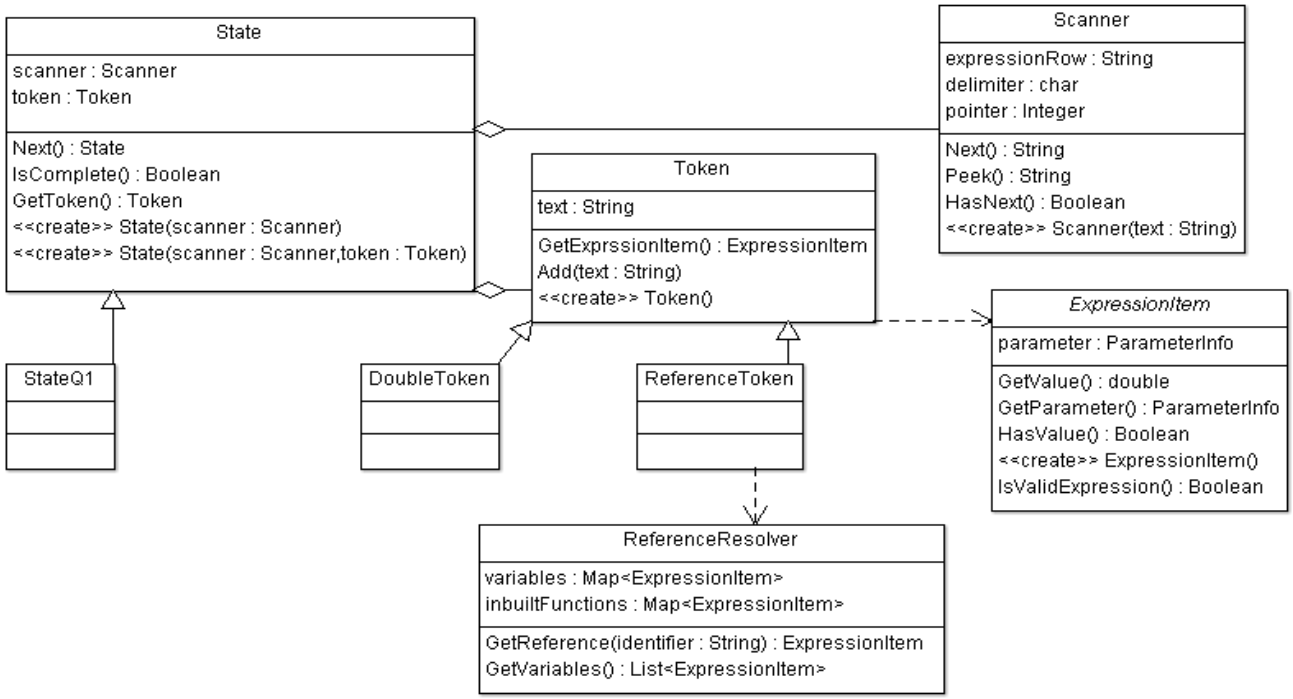
Skanning av symboler

Skanningen av symboler sker genom en finit automata. De tillåtna symbolerna är operatorer, rationella tal och textreferenser. Operatorer identifieras lätt och till dem hör bland annat +, -, *, / och (). Rationella tal består av valfritt antal siffror, eventuellt följt av decimalpunkt och valfritt antal siffror. Detta kan i sin tur följas av ett ‘e’ eller ‘E’, sedan + eller - och slutligen minst en siffra. Textreferenser består av valfritt antal alfanumeriska tecken, varav minst ett alfabetiskt som inte är ‘e’ eller ‘E’. Blanka tecken är tillåtna i ett textuttryck, förutom radbrytningar. Blanka tecken får dock bara förekomma mellan distinkta symboler. Automatan som används som grund för att skanna symboler kan ses i figur 3. Pilar utan symboler betyder vilket tecken som helst.



Figur 3. Automata

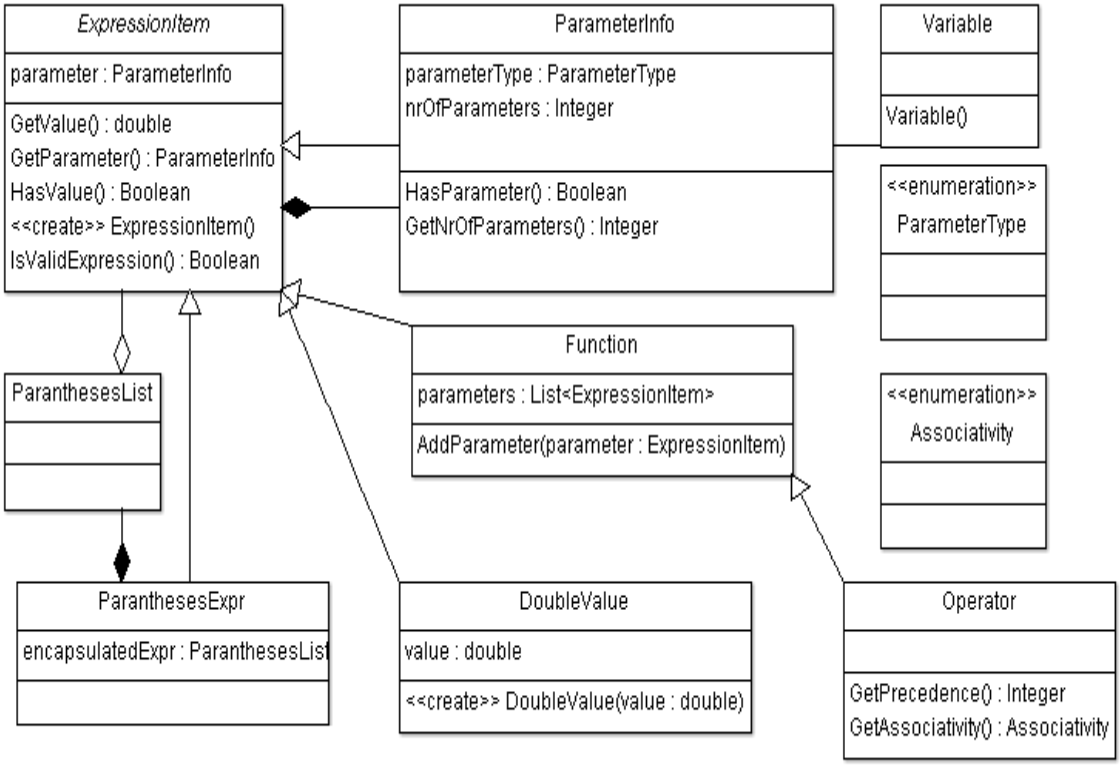
Automatan skapar symboler som en RowEvaluator lägger in i en lista. Hur relationerna mellan klasserna ser ut för skanningen av symbolerna kan ses i figur 4.



Figur 4. Tokenizer

Tolkning av symboler

När en RowEvaluator har skapat en lista med symboler så gås hela listan igenom och varje symbol ansvarar för att skapa en motsvarande ExpressionItem. ExpressionItem är en abstrakt klass som är superklass för alla olika typer av uttrycksdelar. Hur klasser ärver från ExpressionItem kan ses i figur 5.



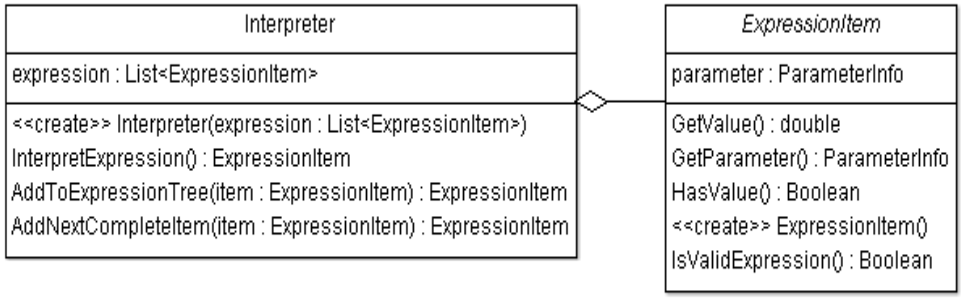
Figur 5. Parser

ExpressionItem har ett antal metoder för att RowEvaluator skall kunna avgöra hur varje ExpressionItem skall hanteras.

Det viktiga som händer när symbolerna tolkas sker i subklasserna till Token och ExpressionItem. Rationella tal får ett värde av typen double, variabler hämtas från ReferenceManager och funktioner hämtas från ReferenceManager. I ReferenceManager lagras objekt som representerar variabler och funktioner. För variabler så skickas referenser tillbaka, medan funktioner skickas som kopior av de lagrade objekten. På så sätt kan variabler lätt erhålla nya värden, och funktioner kan hanteras beroende på vilka parametrar de har.

Tolkning av uttryck

När alla uttrycksdelar har skapats så börjar tolkningen av hela uttrycket. Från listan av ExpressionItem så sätts ett träd ihop i enlighet med matematiska räkneregler. Hur klasserna beror på varandra kan ses i figur 6.



Figur 6. Interpreter

Undantag

Relevanta programdelar kommer att kasta undantag när det krävs. Alla berörda programdelar kommer lägga till information till ett felmeddelande, som huvudfönstret sedan kan visa för användaren.

Tidsplanering