Gymnasiearbete (100 p)
IT-Gymnasiet Göteborg
HT 2017 - VT 2018

Supervisors:
Daniel Berg, David Lundholm

Erwall, a High Level General Purpose Programming Language

28 januari 2018

Created with LaTeX

Author:
Erik Wallström

Abstract

Keywords: Erwall, Programming, Language, C, Compiler

Innehåll

1 Inledning

Jag har valt att skapa ett nytt programmeringsspråk som kan användas praktiskt. Språket kommer att vara kompilerat samt ha statisk typning. Målet är att konstruera en bättre version av det redan existerande programmeringsspråket C. Projektet kommer att ha öppen källkod samt en fri upphovsrättslicens. Jag kommer använda det distribuerade versionshanteringssystemet *git* för att hålla ordning på utveckling och ändring av källkod, samt möjliggöra för andra människor att bidra med förbättringar och ändringar. Detta är ett ganska ambitiöst projekt.

Fokus på programmering och utvecklingen av en kompilator, fokus på språk och standard senare.

Programmering har varit min hobby i nästan 5 år, och under den tiden har jag testat ett stort antal av de existerande programmeringsspråken. Dock så har jag ännu inte hittat något språk som jag anser vara perfekt; antingen saknas funktioner, eller så finns det onödiga och överflödiga funktioner som jag ogillar, med mera. Genom att skapa ett eget programmeringsspråk så kan jag anpassa och förbättra vissa delar så att det blir det perfekta programmeringsspråket för mig.

1.1 Bakgrund

Jag har gått kursen Programmering 1, och går just nu i Programmering 2. Jag har tidigare erfarenhet av många programmeringsspråk samt *git*, samt skapat en interpretator för programmeringsspråket *Brainfuck*. Jag har även konstruerat en textredigerare och utvecklingsmiljö för programmeringsspråket *Lua*, där jag bland annat haft förgmarkeringar för syntax.

1.2 Syfte

Syftet med detta projekt är att i slutändan ha ett fungerande programmeringsspråk som på sikt kan ersätta programmeringsspråket C, som idag fortfarande är ett av de mest använda programmeringsspråket. Dessutom så ska det visa att ett programmeringsspråk inte är något märkvärdigt; många programmerare tänker klagar på programmeringsspråk, men ytterst få försöker förbättra dem eftersom att mentaliteten att man inte ska återuppfinna det som redan finns, samt att man inte ska byta ut det som redan fungerar. Detta leder till att programmeringsspråk får en speciell status, och många tänker inte på att det rör sig om en viss standard som är satt, eller ett program som helt enkelt tolkar och översätter koden du skriver till maskinkod. Många tycker att det är orimligt att skapa ett nytt programmeringsspråk.

1.3 Frågeställningar

Under projektets gång vill jag försöka utforska och svara på dessa frågor som jag hade från början.

- Vilka delar är en kompilator uppbyggd av?

  Alla stora program är uppbyggda av olika delar som tillsammans utför en uppgift. Så vilka delar är det som en modern kompilator för ett språk generellt är uppbyggd av?

- Hur bra fungerar C som ett programmeringsspråk för att skapa en kompilator?

  Debatten om vilka språk som passar för vilka uppgifter har hållit på i många år. Språket som jag har mest erfarenhet av än så länge är C, så hur bra passar det till just det här projektet? Vilka fördelar och nackdelar finns det då det är väldigt nära hårdvaran, samt att standard- biblioteket är minimalt.

- Är C ett bra mål att generera kod till?

  Är det en bra idé att översätta koden skriven i det nya språket till C kod istället för till exemepl maskinkod eller använda virtuella maskiner som till exempel *LLVM*? Vilka fördelar och nackdelar kan det ha?

- Vad krävs av ett programmeringsspråk för att det ska anses som bra"?

  Vad är det egentligen som gör att folk gillar och ogillar språk? Var går balansen mellan bra och användbar? Skaparen av programmeringsspråket C++ har bland annat sagt "Det finns två olika typer av språk; språk som folk klagar på, och språk som inte används".

- Hur ska man planera ett projekt för att tillåta enkel felsökning och tillägg av nya funktioner?

  Stora projekt så som detta kommer vara kommer kräva många ändringar och försvåra felsökning och förbättringar. Vilka verktyg och praxis bör man använda och följa för att förenkla detta?

1.4 Kravspecifikation

- Logisk och konsekvent syntax
- Explicita deklarationer
- Högpresterande och nära maskinvaran

Några specifika specifikationer på språket kommer vara följande:

- Rekursiva omfång (?) med funktioner och kommentarer
- Garanterade svansanrop (?)
- Kompatibelt med C
- Strikt typsystem med riktiga konstanter
- Inget odefinerat beteende
- Listor som första klassens medborgare
- Markerade unioner

  med mera.

1.5 Metod och material

Projektet är skrivet i ren gnu-11 C kod med hjälp av ett textredigeringsprogram (vim) i operativsystemet Arch Linux och kompilerat med kompilatorn GCC. Inga externa bibliotek och API:er har använts, utan allt är skrivet från grunden.

1.6 Teoretisk bakgrund

This subsection's content...

2 Resultatredovisning

This section's content...

3 Diskussion och slutsatser

This section's content...

4 Källförteckning

This section's content...

5 Bilagor

This section's content...