VSA3

Fontys FHICT 2015-2016
klas : 31C
Auteur : Wouter Vanmulken

Inhoudsopgave

Inleiding

Dit verslag gaat over het traject waarin ik, Wouter Vanmulken uit klas 31C op fontys 2016. In opdracht van fontys FHICT c++ heb geleerd.

Opdracht

De orginele opdracht was om c++ te leren doormiddel van de youtube filmpjes van het kanaal van CodingMadeEasy (https://goo.gl/YjIfl9). Gezien ik makkelijk bestuurbare applicaties met een GUI wou maken ben ik hiernaast ook onderzoek gaan doen naar libraries hiervoor. Hierbij heb ik meerdere libraries onderzocht, namelijk Visual C++, QT en Gtk ,waarna ik uiteindelijk het meeste in GTK heb gewerkt.

Motivatie voor de keuze van de opdracht

De reden waarom ik voor C++ heb gekozen voor deze opdracht was voornamelijk dat C++ een zeer gewilde taal is voor werkgevers. C++ kan wanneer goed geschreven multi-platform werken en is niet alleen een van de krachtigste maar ook een van de snelste talen waar vandaag de dag mee wordt ontwikkeld.

Samenvatting

Ik heb eerst de basis van C++ geleerd, daarna ben ik visual c++ gaan bestuderen. Vond het helaas weinig uitdagend. Daarna heb ik Qt uitgeprobeerd en was niet onder de indruk dus ben ik gaan werken op linux en mijn applicatie's gaan richten op linux met Gtkmm (de c++ wrapper voor Gtk). Dit was een flinke uitdaging voornamelijk het werken met de compiler en hierbij de libraries toevoegen was verschrikkelijk. Na vele IDE's te hebben geprobeerd kon ik geen enkele IDE werkend krijgen in linux samen met Gtk. Hierdoor ben ik in een simpele text-editor gaan werken en heb steeds gecompileerd met de hand.

Hierna heb ik twee iets grotere applicatie's gemaakt. De eerste applicatie is een time-tracker, deze kijkt of een bepaalde code die de gebruiker kan aangeven runt op een gegeven moment in tijd. Deze applicatie heb ik gemaakt om goed kennis te maken met de GUI van GTK en treeview wat ontzettend lastig was.

En de tweede applicatie download alle volledige foto's van een imageboard dat 4chan heet wanneer een gebruiker het board-ID en threadnummer meegeeft. Deze applicatie heb ik gemaakt om kennis te maken met het downloaden van files met curl,het lezen en schrijven naar files en het parsen van json in c++.

De grootste bottleneck was helaas het werken met Gtkmm doordat ik dit niet kon compileren, hierdoor moet ik extra tijd steken in het leren van de compiler en heb ik helaas niet altijd even productief kunnen werken.

Het bewijs dat ik de youtube filmpjes heb bekeken en de link naar de github repo zijn aan het einde van het document te vinden.

Traject

(inleiding)

Toen ik begin met het leren van C++ leek het allemaal redelijk te doen binnen 28 uur, maar zoals ik gemerkt heb over de laatste ongeveer 30 uur is het erg weinig, zeker wanneer we het hebben om een taal zo complex als c++ met GUI libraries nog complexer.

(Begin leren C++)

Ik ben als allereerste begonnen met het kijken van de eerste ongeveer 40 filmpjes van de bovengenoemde playlist. Hierna had ik het gevoel dat ik wel een redelijke applicatie kon schrijven. Dit had ik flink mis. C++ heeft namelijk een andere structuur qua klassen opbouwen, alles word namelijk gedeclareerd in de .h files en de echte code staat dan in een .cc of .cpp file. De compiler is in staat om dit allemaal bij elkaar te zoeken en zo in elkaar te zetten dat het kan draaien. Het filesysteem van c++ was mijn eerste kleine horde die ik moest overkomen, gelukkig nog een kleine horde in vergelijking met wat nog ging komen maar een die toch wel wat onderzoek in is gaan zitten. Daarna kwam het correct aanroepen van namespaces, pointers en objecten, dit bleek al een stuk moeilijker. Dit was initieel ontzettend lastig maar na met iemand gesproken te hebben die c++ al wat langer kende was ik weer op weg.

(Visual c++ & Qt)

Toen heb ik een paar vrij “Hello World” programmaatjes geschreven en een paar simpele programma’s die labels veranderde wanneer er op een button werd gedrukt of een MessageBox opende. Helaas zijn deze verloren gaan bij mijn immigratie naar linux. Toen ik een tijdje bezig was realiseerde ik me dat visual c++ erg leek op windows form applicaties in c# en leek me dit niet echt een uitdaging om verder te onderzoeken voor VSA. Ik ga dit zeker nog een keer onderzoeken naast school maar ik wou gelieve voor VSA iets leren wat me echt zou uitdagen. En dat heeft het ten zeerste gedaan. Na op windows ook nog de Qt library te hebben geprobeerd, en een applicatie gemaakt te hebben die labels aanpaste aan array elke klik voelde deze traag aan en ben ik verder gaan zoeken naar alternatieven.

(Gtkmm)

Toen ben ik gaan kijken Gdx of eigenlijk de wrapper van Gdx voor C++ namelijk Gdxmm. Na de overstap op linux te hebben gemaakt was ik klaar om de strijd aan te gaan met de documentatie van Gdxmm...die niet werkte. Ik heb vrijwel elke IDE geprobeert maar geen enkele kreeg ik goed werkend samen met gdxmm. Hier is helaas vrij weinig over te vinden, ik heb urenlang zitten zoeken naar een oplossing van dit probleem, en na vele uren en dan bedoel ik ook vele uren, menig IDE’s en vele verslagen buien ben ik met een text editor en een compiler gaan werken.
Dit was een compleet nieuwe ervaring en ook die volgens mij zeker weten ervoor heeft gezorgd dat ik commando’s beter uit mijn hoofd ken en precieser ben gaan werken. Maar de eerste paar momenten waren eenhele aanpassing, het is een totaal andere ervaring om in een text editor te gaan werken zonder intellisense omdat er weinig feedback is op wat je intypt. Elke keer als ik moest compileren moest ik weer een speciaal commando runnen en die dan met de output gaan verbeteren.

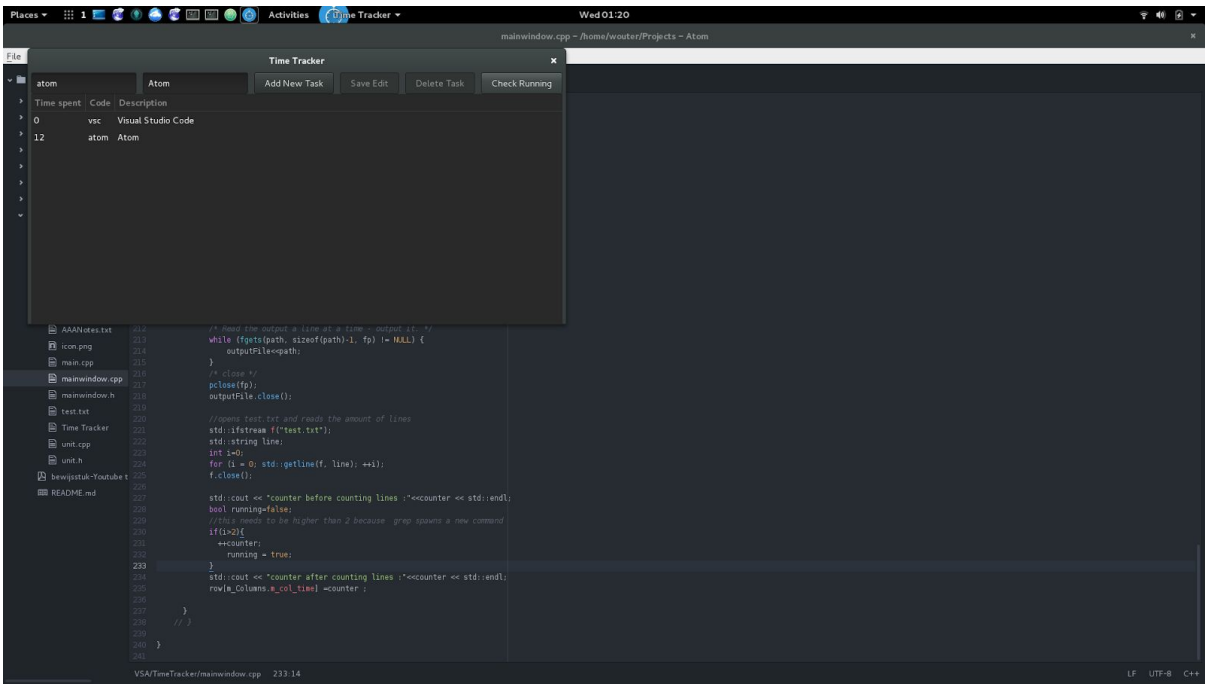
Hoewel dit initieel erg lastig was, maakte mij dit nauwkeuriger en ging ik dingen beter onthouden. En juist omdat alles meer moeite koste, ging ik er ook meer op letten wat ik precies schreef en naar mate van tijd werden spelfouten in de code ook steeds minder.

(Gtkmm applicaties bouwen)

Toen ik eindelijk mijn gtkmm applicaties kon compileren, ging er ineens een wereld voor me open. Initieel had ik glade gebruikt, wat een gui bouwer is, om mijn gui's te bouwen maar merkte al snel dat dit minder controle gaf over wat je toe kon voegen dan wanneer het in code werd toegevoegd. Daarom ben ik toen over gegaan naar het handmatig maken van de GUI wat erg beviel. Het is namelijk mogelijk om boxes toe te voegen, die een horizontale of verticale layout te geven en hier dan widgets in te plaatsen.

Omdat ik graag wou laten zien wat ik nou kon met gtkmm heb ik een tijd tracking programma gemaakt. Dit gebruikt een linux command om te kijken of een programma runt. Dit programma zal dan ook niet op windows werken gezien dit een specifiek commando voor linux is. Hoewel hierbij natuurlijk ook nog code bij zou kunnen om het windows compatible te maken. De applicaties zijn allemaal terug te vinden op de github pagina (deze is terug te vinden aan het einde van het document.)

hieronder kun je de timetracker zien.

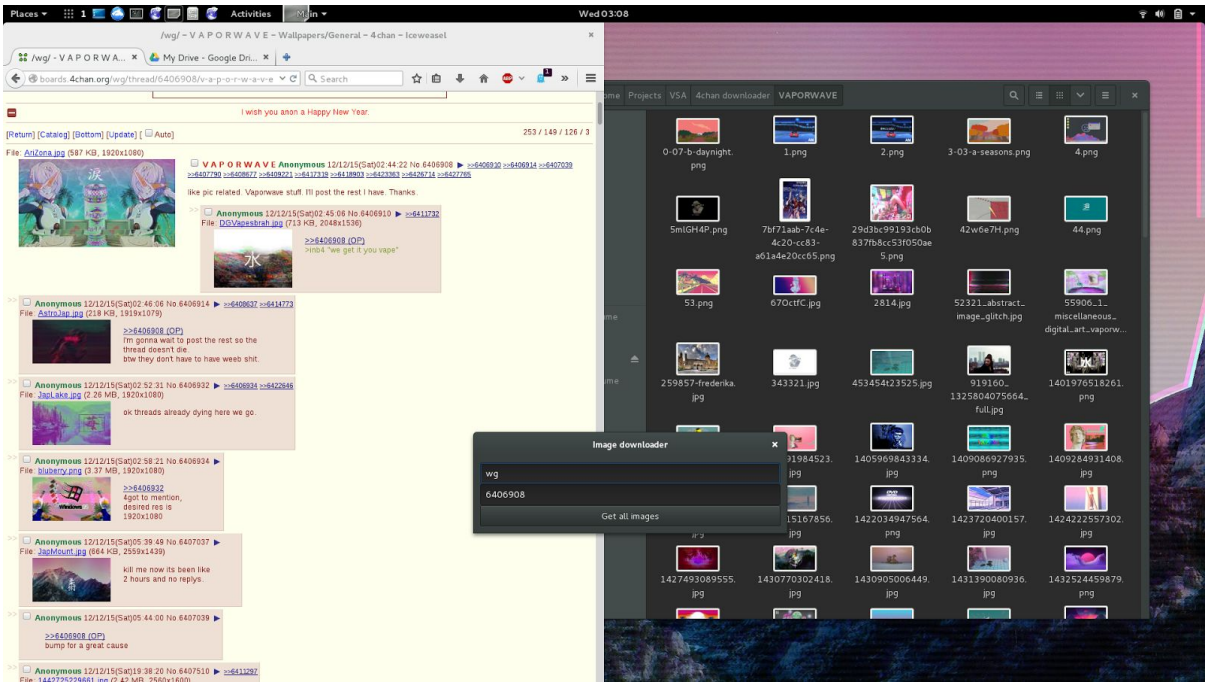


De applicatie gebruikt de knop om te kijken of er een van de aangegeven programma's aan het werk is en telt hier dan 1 bij op. Om deze applicatie te maken heb ik een voorbeeld gebruikt om de treeview werkend te maken.
Zoals je kunt zien is de timer van Atom omhoog gegaan omdat die draait terwijl die van VIsual Studio Code op 0 is blijven staan omdat die niet draait.

Daarnaast heb ik ook nog een 4chan.org image downloader gemaakt die alle foto's van een bepaalde thread afhaalt met de orginele naam en extensie en ze in een map plaatst met de

naam van de Thread. Dit maakt gebruik van de 4chan api en gebruikt de curl library om files op te halen en rapidJson om de json te parsen.

Hieronder kun je de applicatie zien, en zien dat het ook werkt. Zoals je kunt zien zijn de juiste plaatjes in de correcte folder geplaatst.



Competenties & Conclusie

In het begin van de vsa heb ik de onderstaande competenties ingevuld om aan te werken :

- Analyseren
- Professionele toepassing
- Ontwerpen
- Realiseren

Tijdens het leren van C++ heb ik me voornamelijk gefocused op het analyseren en realiseren van applicaties. Doordat ik niet echt een voorgekauwd traject heb gevolgd zoals dat op school word gegeven, moest ik veel documentatie analyseren en daarna naar echte applicaties realiseren. DIt zorgde ervoor dat ik ook voornamelijk heb geleerd hoe ik een nieuwe taal zelf leer, zonder hulp. Dit is zeer belangrijk gezien talen constant veranderen en er elk jaar weer nieuwe talen en technologiën bijkomen, terwijl ik niet altijd op school bij kan gaan leren.

Ik denk dat ik ontzettend veel heb geleerd tijdens dit project en daarbij voornamelijk geleerd heb hoe ik zelf moet leren. Daarom denk ik dat dit een zeer geslaagd traject was. Hoewel ik best wel lastig vond om VSA naast school nog erbij te maken, vond ik het wel goed dat we dit vak krijgen. Door dit soort vakken krijgen we echte ervaringen hoe het gaat wanneer we van school af zijn. Wat ons beter voorbereid op de echte wereld.

Daarom vind ik dit een zeer geslaagd vak, en vond het erg leuk om C++ te leren.

links

Youtube playlist CodingMadeEasy:
https://www.youtube.com/watch?v=JX14ObbbCPI&list=PL2DD6A625AD033D36

Github Repository :
https://github.com/woutie012006/VSA/

Github Release :
https://github.com/woutie012006/VSA/releases/tag/1.0

Bewijs van het kijken van de youtube playlist
https://github.com/woutie012006/VSA/blob/master/bewijsstuk-Youtube%20tutorials.pdf