TEHNIČKA ŠKOLA RUĐERA BOŠKOVIĆA  
GETALDIĆEVA 4, ZAGREB

ZAVRŠNI STRUČNI RAD:

CreateUI

MENTOR: Goran Gomaz UČENIK: Roko Burilo  
 RAZRED: 4.H

Zagreb, travanj 2020.

1. Uvod

CreateUI je završni rad softverskog tipa čiji je cilj osmisliti i implementirati framework za stvaranje i upravljanjem GUI sustava. Razlog odabiru ovog rada je bilo u stjecanju prakse u izradi velikih softverskih projekata. Različiti manji dijelovi ovog projekta mogu se gledati i samostalno jer su i to projekti vrijedni spomena.

Tijekom izrade ovog projekta izradio se:

* Svojevrsni jezik koji opisuje strukturu scena
* Svojevrsni jezik koji opisuje stilove GUI elemenata (Block-ova)
* Parser[[1]](#footnote-1) za ta dva jezika
* Dizajn sučelja između CUI-ja i različitih *binding-a[[2]](#footnote-2)*
* Binding CUI-ja sa multimedijskom bibliotekom SFML zvan CUI-SFML
* Nekoliko malih primjera mogućih GUI programa u CUI-SFML biblioteci
* Parsiranje tijekom kompilacije i primitivan oblik rukovanje parsing greškama

U projektu se koristio C++17 zbog olakšavanja restrikcija na *constexpr* uvjetima rada, Tessilova implementacija rječnika zvan tsl::hopscotch\_map, SFML zbog lakog ugrađivanja biblioteke i jednostavnog sučelja prema grafici.

Obje biblioteke (CUI i CUI-SFML) su oblika *header-only* zbog lakog prototipiranja i ugrađivanja u druge projekte.

1. CUI framework
2. Struktura CUI sustava

Strukturu CUI sustava čine podsustavi Styles i Scenes.

Styles određuje stilove čije atribute definira korisnik, tj. pisac stila. Ti stilovi se onda mogu iskoristiti u Scenes sustavu, gdje jedan GUI blok može definirati stilove koje će koristiti.

Scenes sustav određuje samu strukturu, tj. odnos članova scenskog grafa. On definira pojedine značajke svakog člana, kao što su stilovi i tekst.

Ta dva sustava spaja SceneGraph klasa skupa sa Node klasom. Node klasa sadrži atribute i pravila, dok SceneGraph sadržava Node-ove.

Scenes

Styles

SceneGraph

Node

Implementacija  
prozora

1. Styles

Styles je podsustav CUI-a i format koji definira stilove koje onda može Scenes podsustav iskoristiti za svoje blokove.

Mogući atributi su:

* background – postavlja pozadinu na boju ili sliku; Moguće vrijednosti:
  + rgb(crvena, zelena, plava)
  + rgba(crvena, zelena, plava, alpha)
  + url(put\_to\_datoteke)
  + red
  + green
  + blue
  + yellow
  + transparent
  + white
  + black
* x – postavlja poziciju bloka na x osi; Moguće vrijednosti:
  + 50 – postavlja blok na pixelu 50 gledano slijeva
  + 30.45% - adaptira poziciju na x osi ovisno o duljini roditelja
  + left – postavlja blok na lijevu stranu roditelja
  + right – postavlja blok na desnu stranu roditelja
  + center – postavlja blok u sredinu roditelja
* y – postavlja poziciju bloka na y osi; Moguće vrijednosti:
  + 50 – postavlja blok na pixelu 50 gledano slijeva
  + 30.45% - adaptira poziciju na y osi ovisno o visini roditelja
  + top – postavlja blok na vrh roditelja
  + bottom – postavlja blok na dno roditelja
  + center – postavlja visinu bloka u sredinu visine roditelja
* width – postavlja duljinu bloka; Moguće vrijednosti:
  + 120 – postavlja duljinu bloka na 120 pixela
  + 50% - postavlja duljinu bloka na pola duljine roditelja
* height – postavlja visinu bloka; Moguće vrijednosti:
  + 120 – postavlja visinu bloka na 120 pixela
  + 50% - postavlja visinu bloka na pola visine roditelja
* font – postavlja font teksta; Moguća vrijednost:
  + url(put\_to\_datoteke)
* font\_size – postavlja veličinu fonta / teksta na zadanu vrijednost; Moguća vrijednost:
  + 30 – postavlja veličinu fonta na 30
* text\_color – postavlja boju teksta; Moguća vrijednost:
  + rgb(crvena, zelena, plava)
  + rgba(crvena, zelena, plava, alpha)
  + red
  + green
  + blue
  + yellow
  + transparent
  + white
  + black
* text\_position – postavlja gdje će tekst biti unutar bloka; Moguće vrijednosti:
  + TopLeft
  + Top
  + TopRight
  + Left
  + Center
  + Right
  + BottomLeft
  + Bottom
  + BottomRight

1. Svrha

Styles podsustav stvoren je s razlogom da pruža stilove u stalnom, definiranom paketu kako bi ga Scenes podsustav mogao iskoristiti da definira atribute blokova. Sa Styles-om nije potrebno definirati stil pojedinačno za svaki blok, već svaki blok može samo uputiti CUI sustavu koji stil želi iskoristiti te, ako postoji, postaviti atribute bloka na atribute stila. Ako stil mijenja iste atribute kao prijašnji stil, atribut će se postaviti na atribut stila koji je novijeg stila.

Situacija se može lako predočiti sljedećom analogijom:   
Zamislimo da smo na servisu za bojanje automobila i zamislimo da su dijelovi našeg automobila, kao što su gume, krov, vozačeva vrata i suvozačeva vrata, njegovi atributi. I zamislimo da su stilovi naše naredbe za bojanje dijelova našeg auta. Te naredbe moramo reći serviseru slijedno te će ih on tako i izvršiti. Zatim zamislimo da smo zadali naredbu da oboja gume u plavo, a krov u crveno. I onda da smo mu zadali da nam vozačeva vrata i suvozačeva vrata oboja u zeleno, ali da gume oboja u crno.

Dogodit će se sljedeće:

Serviser će izvršiti našu prvu naredbu i obojati gume u plavo te krov u crveno. Zatim će izvršiti našu drugu naredbu i obojati vozačeva vrata i suvozačeva vrata u zeleno, ali će prebojiti gume te će gume postati crne.

1. Format

Styles format je jako sličan CSS-ovom formatu datoteke, ali se ipak razlikuje u određenim stavkama. CSS nema pojam o korisničko-definiranim događajima, ali Styles ima. To jest, ima sučelje gdje korisnik može definirati na koje događaje želi postaviti stil.

Općeniti format datoteke je:

ime-stila ?[ime-događaja1 ?, ime-događaja2, …?]? {

ime-atributa: vrijednost-atributa;

…

}

…

gdje ?...bilošto…? označava neobavezan dio formata i gdje je:

* ime-stila – alfanumerička vrijednost koja sadrži znakove od [A-Za-z0-9\_] počevši sa slovom ili podvlakom
* ime-događaja – niz znakova koji predstavljaju ime događaja
* ime-atributa – niz znakova koji predstavljaju ime atributa
* vrijednost-atributa – može biti cjelobrojni ili decimalni broj, funkcija ili riječ
  + primjer:
    - 5
    - 3.14
    - rgba(255, 125, 0, 225)
    - green

1. Parsiranje formata

Iz formata imamo sve informacije koje nam trebaju da stvorimo parser tog formata. Parsiranje je, u osnovi, iteriranje kroz neki skup podataka i provođenje pravila formata te izvlačenje tokena iz tog skupa podataka. U slučaju CUI-a, taj skup podataka je ugrađen niz znakova iz neke zadane datoteke tijekom kompilacije.

U pseudokodu:

parsiraj(ulaz):

dok nije kraj ulaza:

pojedi tok podataka dok ne naiđeš na bilokoji od sljedećih znakova: []:;{}

makni razmake i slično iz pojedenog

ovisno na kojemu je znaku prekinuto jedenje:

ako je na '[':

ako prijašnje stanje nije početak stila, baci grešku

ako je obrađen podatak prazan, baci grešku

ako nije valjano ime, baci grešku

dodaj definiciju

postavi ime zadnje definicije na obrađen podatak

promijeni prijašnje stanje jedenja

ako je na ']':

ako prijašnje stanje nije početak liste događaja, baci grešku

ako je obrađen podatak prazan, baci grešku

ako se dogodila greška tijekom parsiranja liste događaja, baci grešku

promijeni prijašnje stanje jedenja

ako je na '{':

ako prijašnje stanje nije kraj liste događaja i nije početak stila, baci grešku

ako je prijašnje stanje početak stila i ako je obrađen podatak prazan, baci grešku

ako se dogodila greška tijekom parsiranja liste događaja, baci grešku

ako je prijašnje stanje početak stila:

ako nije valjano ime stila, baci grešku

dodaj definiciju

postavi ime zadnje definicije na obrađen podatak

promijeni prijašnje stanje jedenja

ako je na '}':

ako prijašnje stanje nije početak definicije stila, tj. '{' i nije kraj definicije atributa, baci grešku

promijeni prijašnje stanje jedenja

ako je na ';':

ako prijašnje stanje nije početak vrijednosti atributa, baci grešku

ako je obrađen podatak prazan, baci grešku

postavi vrijednost atributa zadnje definicije na obrađen podatak

promijeni prijašnje stanje jedenja

ako je na ':':

ako prijašnje stanje nije početak definicije stila i nije kraj vrijednosti atributa, baci grešku

ako je obrađen podatak prazan, baci grešku

postavi vrijednost atributa zadnje definicije na obrađen podatak

promijeni prijašnje stanje jedenja

vrati tokenizirani rezultat

1. Primjeri datoteka

bgcolor\_red {

background: white;

background: rgba(60, 40, 50, 100);

background: rgb(255, 0, 0); // Vrijednost se postavi na zadnji postavljeni atribut u stilu

}

button {

background: url(./abc.png);

text\_color: red;

font: url(./arial.ttf);

font\_size: 50;

text\_position: center;

x: center;

y: center;

width: 160;

height: 90;

}

1. engl. parser – inform. program koji obavlja sintaktičku analizu nekog jezika; sintaktički analizator [↑](#footnote-ref-1)
2. engl. binding – u kontekstu softverskih biblioteka označuje povezivanje različitih platformi ili jezika na biblioteku [↑](#footnote-ref-2)