

Електротехнички факултет Универзитета у Београду  
Катедра за Рачунарску технику и информатику



**SeeGL – Софтверски алат за учење графичке библиотеке OpenGL**

**Прилог Г - Упутство за употребу алата SeeGL**

Марко Првуловић и Ђорђе Ђурђевић

# Садржај

Г-1. Увод.....	3
Г-2. Кориснички интерфејс алата .....	4
Г-2.1. Интеракција са прозором са наредбама актуелног OpenGL програма .....	4
Г-2.2. Интеракција са прозором за приказ ЕРС шеме .....	7
Г-2.3. Интеракција са прозором за преглед атрибута.....	9
Г-2.4. Операције над датотеком.....	10
Г-2.5. Прозор за приказ и дефинисање променљивих .....	11
Г-2.6. Мени и палете алата.....	12
Г-3. Примери употребе програма .....	13
Г-3.1. Вежба 1 - Састављање једноставног OpenGL програма .....	14
Г-3.2. Вежба 2 – Анализа програма .....	16
Г-3.3. Вежба 3 – Едитовање OpenGL програма .....	18
Г-3.4. Вежба 4 – Интерактивне могућности над приказом ЕРС шеме .....	18

## Г-1. Увод

Овај документ представља упутство за употребу софтверског алата *SeeGL*. Најпре су описани елементи корисничког интерфејса програма. Затим су описани примери употребе програма кроз 4 вежбе које демонстрирају могућности овог алата. Прва вежба демонстрира састављање једноставног програма, његово извршавање и добијање резултата цртања сцене. Друга вежба демонстрира анализу програма, односно могућности извршавања програма и прегледа стања коначног аутомата. Трећа вежба демонстрира могућности едитовања програма. Четврта вежба демонстрира интерактивне могућности приказа шеме ЕРС модела.

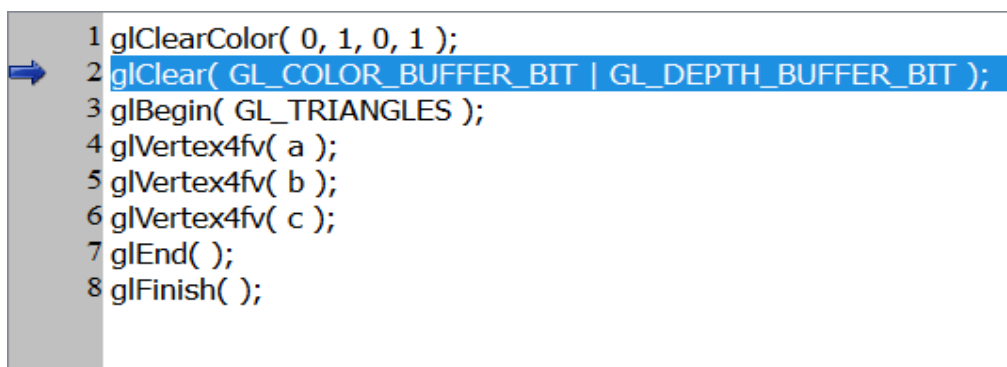
## Г-2. Кориснички интерфејс алата

У овом одељку описани су прозори апликације са могућим акцијама у њима, главни мени и палета алата (енг. *toolbar*). Сви прозори, сем прозора са *OpenGL* програмом, имају могућност да буду независни прозори или уклопљени у главни прозор, усидредни (енг. *docked*) уз његову ивицу (горе, лево, десно или доле). Графички интерфејс алата *SeeGL* приказује следеће прозоре:

- Прозор са наредбама актуелног *OpenGL* програма.
- Прозор са приказом ЕРС шемом (“*EPC Window*”) која моделира ток обраде података ове библиотеке.
- Прозор са листом атрибута за преглед (“*Watch Window*”).
- Прозор са уведеним променљивама (“*Variables Window*”), које се могу адресирати у позивима наредби у актуелном програму.
- Прозор са резултатом цртања сцене (“*Rendering Window*”).

### Г-2.1. Интеракција са прозором са наредбама актуелног *OpenGL* програма

У прозору се приказују наредбе *OpenGL* програма. Посебно је стрелицом означена наредба која је на реду за извршење, а селектована је наредба на коју се односи акција едитовања. На слици 1 је приказан овај прозор.









```
1 glClearColor( 0, 1, 0, 1 );
2 glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
3 glBegin( GL_TRIANGLES );
4 glVertex4fv( a );
5 glVertex4fv( b );
6 glVertex4fv( c );
7 glEnd( );
8 glFinish( );
```

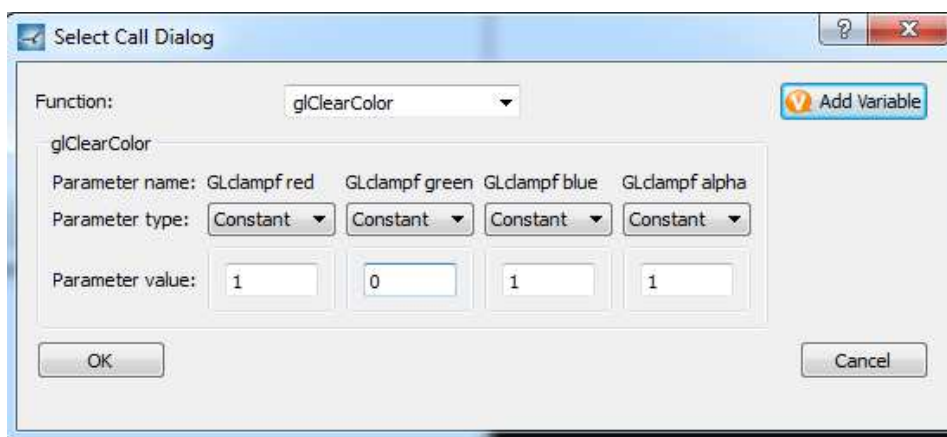
Слика 1 – Централни прозор алата који приказује актуелни програм.

Акције које су могуће над овим прозором су објашњене у наставку:




1. Обележавање наредбе или региона. Наредба или регион се обележава левим кликом миша на текст наредбе или региона. Обележена наредба или регион представља референтну позицију за следеће акције: додавање наредбе, избацивање наредбе, измена наредбе, ручно писање наредбе, померање наредбе у смеру на горе/доле, постављање/уклањање позиције заустављања.
2. Обележавање секвенце наредби или региона. Прво се обележи наредба региона која представља почетак или крај секвенце која се жели обележити. Секвенца се обележава тако што се држи притиснут тастер “шифт” (енг. *shift*) и левим кликом миша одабере наредба која представља други крај секвенце у односу на прву обележену наредбу или регион. На платформи Linux уочена је грешка у Qt имплементацији графичке компоненте

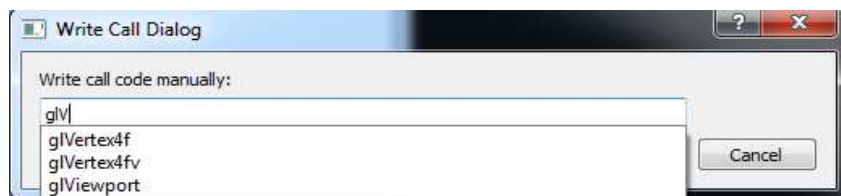
које се манифестује разним проблемима приликом обележавања секвенце наредби користећи тастер “шифт”. Зато је и омогућена алтернатива са тастером “размак” (енг. *space*). На Windows платформи је могуће и са једним и са другим тастером обележити секвенцу наредби. Обележена секвенца се користи приликом стварања региона. Региони могу бити угнеждени.

3. Извршавање наредби до наредне тачке заустављања. Ову акцију је могуће покренути из главног менија преко ставке “Debug->Continue” и кликом на дугме () у палети алата. Уколико коначни аутомат библиотеке није активан, ова акција само активира коначни аутомат и позиционира показивач текуће наредбе на прву наредбу. У супротном извршава редом наредбе почевши од наредбе на коју указује текући показивач до наредбе означене тачком заустављања. Уколико нема наредбе означене тачком заустављања програм се извршава до краја.
4. Извршавање наредне наредбе. Ову акцију је могуће покренути из главног менија преко ставке “Debug->Next” и кликом на дугме () у палети алата. Уколико коначни аутомат библиотеке није активан, ова акција само активира коначни аутомат и позиционира показивач текуће наредбе на прву наредбу. У супротном извршава наредбу на коју указује показивач текуће наредбе.
5. Извршавање до наредбе за цртање сцене (glFinish или glFlush). Ову акцију је могуће покренути из главног менија преко ставке “Debug->Next Flush” и кликом на дугме () у палети алата. Уколико коначни аутомат библиотеке није активан, ова акција само активира коначни аутомат и позиционира показивач текуће наредбе на прву наредбу. У супротном извршава редом наредбе почевши од наредбе на коју указује текући показивач до наредбе glFinish или glFlush. Уколико нема наредбе glFinish или glFlush програм се извршава до краја програма.
6. Постављање позиције заустављања програма. Ову акцију је могуће покренути кликом на дугме () из палете алата, из главног менија преко мени ставке “Debug->Breakpoint” или левим кликом на леви део садржаја прозора са програмом (део сиве боје који садржи бројеве линија). У сва три случаја, покретање акције или поставља позицију заустављања или уклања позицију заустављања са одговарајуће наредбе. Позиција се поставља уколико наредби претходно није била додељена позиција заустављања. Позиција се уклања уколико је наредби претходно била додељена позиција заустављања.
7. Заустављање рада коначног аутомата. Ову акцију је могуће покренути кликом на дугме () из палете алата, или из главног менија преко мени ставке “Debug->Reset”. Покретањем ове акције ресетује се коначни аутомат.
8. Додавање нове наредбе. Ову акцију је могуће покренути кликом на дугме () из палете алата, кликом на ставку “Calls->Insert Call” из главног менија, и кликом на истоимену ставку из менија отвореног десним кликом на неку наредбу у програму. Покретање акције отвара нови дијалог, приказан на слици 2. Нова наредба се смешта на позицију између обележене наредбе и наредбе која следи након обележене наредбе.




Слика 2 – Дијалог за дефинисање наредбе (позива функције *OpenGL* библиотеке)

9. Избацивање наредбе. Ову акцију је могуће покренути кликом на дугме (  ) из палете алата, кликом на ставку “Calls->Remove Call” из главног менија, и кликом на истоимену ставку из менија отвореног десним кликом на неку наредбу у програму. Покретање акције уклања наредбу која је обележена. Обележена наредба постаје следећа наредба.
10. Измена наредбе. Ову акцију је могуће покренути кликом на дугме (  ) из палете алата, кликом на ставку “Calls->Modify Call” из главног менија и кликом на истоимену ставку из менија отвореног десним кликом на неку наредбу у програму. Покретање акције отвара исти дијалог (слика 2) као и приликом додавања позива са тим да су наредба и вредности параметара узети од обележене наредбе. Уколико корисник прихвати измене, измењена наредба ће заменити обележену наредбу.
11. Ручно писање наредбе. Ову акцију је могуће покренути кликом на дугме (  ) из палете алата, кликом на ставку “Calls->Write Call” из главног менија, и кликом на истоимену ставку из менија отвореног десним кликом на неку наредбу у програму. Покретање акције отвара дијалог у коме је могуће ручно (преко тастатуре) унети нову наредбу. Она ће бити уметнута између обележене наредбе и наредбе која следи након обележене наредбе. Кориснику је обезбеђено аутокомплетирање имена наредби. На слици 3 је приказан дијалог где корисник ручно уноси позив преко тастатуре.



Слика 3 – Дијалог где корисник ручно уноси позив.

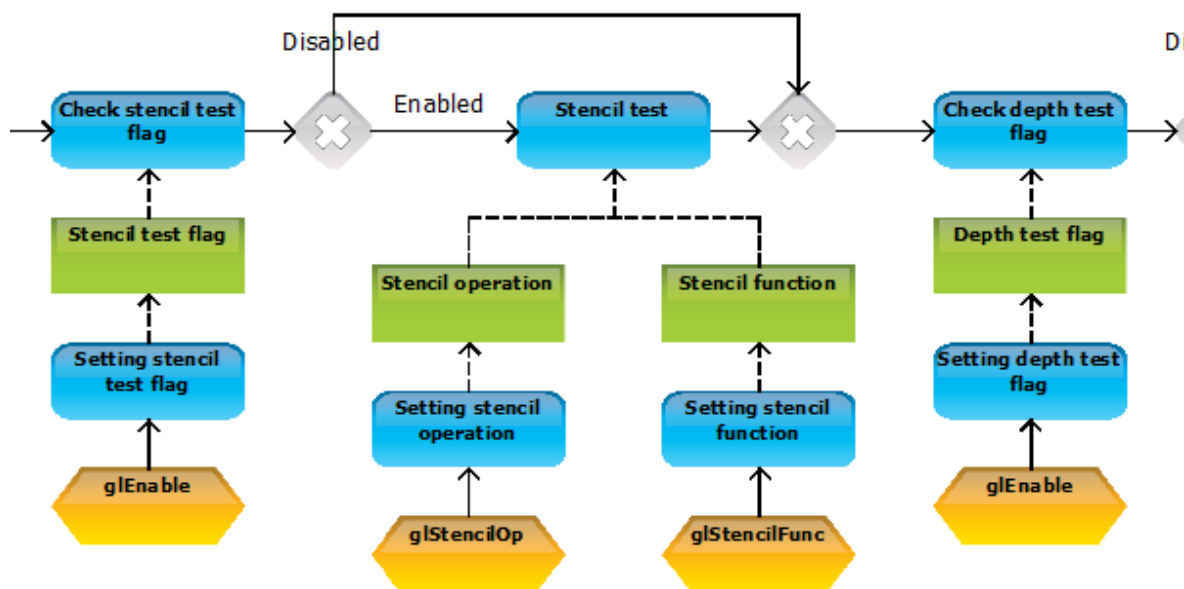
12. Померање наредбе у смеру на горе. Ову акцију је могуће покренути кликом на дугме (  ) из палете алата, кликом на ставку “Calls->Move Call Up” из главног менија, и кликом на истоимену ставку из менија отвореног десним кликом на неку наредбу у програму. Покретање акције замењује места обележеној наредби и наредби која јој

претходи. Уколико је обележена наредба прва у програму, ова акција је онемогућена.

13. Померање наредбе у смеру на доле. Ову акцију је могуће покренути кликом на дугме (↓) из палете алата, кликом на ставку “Calls->Move Call Down” из главног менија и кликом на истоимену ставку из менија отвореног десним кликом на неку наредбу у програму. Покретање акције замењује места обележеној наредби и наредби која јој следи. Уколико је обележена наредба последња у програму, ова акција је онемогућена.
14. Формирање региона од секвенце наредби. Ову акцију је могуће покренути кликом на дугме (📄) из палете алата и кликом на ставку “Calls->Region” из главног менија. Ова акција је омогућена само ако је корисник обележио неку секвенцу наредби. Покретање ове акције отвара дијалог у коме је могуће унети назив региона. Пошто корисник унесе назив региона, обележена секвенца наредби ће се сакрити и уместо ње биће приказан само назив региона.
15. Расформирање региона у секвенцу наредби. Ову акцију је могуће покренути кликом на дугме (✗) из палете алата и кликом на ставку “Calls->Unregion” из главног менија. Ова акција је омогућена само ако је обележен регион. Покретање ове акције уклања регион и враћа наредбе које је регион обухватао у оригиналну секвенцу.

## Г-2.2. Интеракција са прозором за приказ ЕРС шеме

Прозор приказује шему тока обраде података библиотеке *OpenGL*. Користи се графичка нотација ЕРС (Event-driven Process Chaining). Нотација је детаљно приказана у тексту рада. На слици 4 је приказан прозор са делом шеме који је под контролом наредби `glStencilOp` и `glStencilFunc`.



Слика 4 - прозор за приказ ЕРС шеме - део шеме који је под контролом наредби `glStencilOp` и `glStencilFunc`.

Акције које су могуће над овим прозором су објашњене у наставку:

1. Увећавање приказа шеме (енг. *zoom in*). Ову акцију је могуће покренути ротирањем

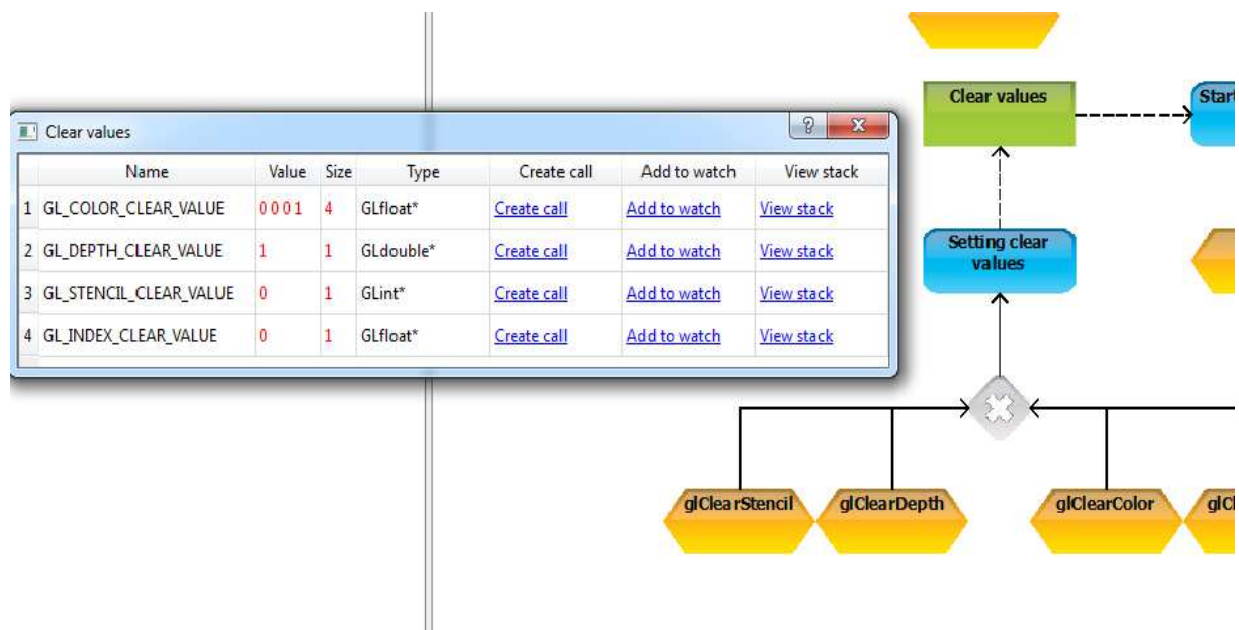
точкића (трећег “дугмета”) миша унапред, кликом на ставку “Zoom In” из менија отвореног десним кликом на неискоришћен простор на шеми или притиском на тастер “+”. Покретање ове акције увећава приказ шеме.

2. Умањивање приказа шеме (енг. *zoom out*). Ову акцију је могуће покренути ротирањем точкића (трећег “дугмета”) миша уназад, кликом на ставку “Zoom Out” из менија отвореног десним кликом на неискоришћен простор на шеми или притиском на тастер “-”. Покретање ове акције умањује приказ шеме.
3. Приказ целе шеме. Ову акцију је могуће покренути кликом на ставку “View All” из менија отвореног десним кликом на неискоришћен простор на шеми. Покретање ове акције умањи приказ шеме толико да се сваки елемент види на приказу шеме.
4. Фокусирање приказа шеме на наредбу која је последња извршена. Ову акцију је могуће покренути кликом на ставку “View Source” из менија отвореног десним кликом на неискоришћен простор на шеми. Уколико није извршена ниједна наредба од последњег активирања коначног аутомата, ова акција је онемогућена. Покретање ове акције фокусира приказ шеме на елемент догађаја који представља наредбу која је последња позвана.
5. Додавање наредбе у програм. Ову акцију је могуће покренути кликом на ставку “Insert <naredba>” из менија отвореног десним кликом на елемент догађаја који представља наредбу коју корисник жели да дода у програм. Параметар <naredba> је назив наредбе која може имати и предефинисану вредност параметара. Уколико постоји наредба са предефинисаним вредностима параметара, покретање ове акције додаће наредбу у програм. У супротном, покретање ове акције отвориће дијалог идентичан ономе за додавање нове наредбе.
6. Преглед атрибута коначног аутомата. Ову акцију је могуће покренути кликом на елемент информационог објекта који представља скуп сродних атрибута коначног аутомата. Покретањем ове акције отвара се нови дијалог са табелом сродних атрибута коначног аутомата. У пододељку Г2.2.1 је објашњена интеракција са овим дијалогом.

### **Г-2.2.1. Интеракција са дијалогом сродних атрибута коначног аутомата**

Пример дијалога са атрибутима коначног аутомата, отворен преко акције 6 из одељка Г2.2 приказан је на слици 5.





Слика 5 – Дијалог средњих атрибута (GL\_COLOR\_CLEAR\_VALUE, GL\_DEPTH\_CLEAR\_VALUE, GL\_STENCIL\_CLEAR\_VALUE, GL\_INDEX\_CLEAR\_VALUE) коначног аутомата, добијен левим кликом миша на елемент “Clear values” на EPC шеми

Акције које су могуће над овим дијалогом објашњене су у наставку:

1. Додавање наредбе у програм. Ову акцију је могуће покренути кликом на дугме “Create Call”. Покретање ове акције отвара дијалог за додавање нове наредбе у програм при чему је у дијалогу већ одабрана наредба која је задужена за контролу атрибута за кога је кликнуто дугме “Create Call”.
2. Додавање атрибута у прозор за преглед атрибута. Ову акцију је могуће покренути кликом на дугме “Add to watch”. Атрибут за кога се кликне дугме “Add to watch” се додаје у прозор за преглед атрибута.
3. Преглед стека атрибута. Ову акцију је могуће покренути кликом на дугме “View Stack”. Покретање ове акције отвара дијалог где је могуће прегледати стек атрибута.

### Г-2.3. Интеракција са прозором за преглед атрибута

Прозор за преглед атрибута приказан је на слици 6.

	Name	Value	Size	Type
	GL_COLOR_CLE...	0 0 0 1	4	GLfloat*
	GL_DEPTH_CLE...	1	1	GLdouble*

Слика 6 – Прозор за преглед атрибута (GL\_COLOR\_CLEAR\_VALUE, GL\_DEPTH\_CLEAR\_VALUE,) коначног аутомата

Акције које су могуће над овим прозором објашњене су у наставку:

1. Додавање атрибута у прозор. Ову акцију је могуће покренути кликом на дугме () из палете алата, кликом на ставку “Watch->Add Watch” из главног менија, или дуплим кликом на неискоришћен простор овог прозора. Покретање ове акције отвара дијалог где је могуће изабрати атрибут који жели да се посматра. Дијалог нуди кориснику могућност аутокомплетирања уколико корисник жели ручно да унесе име атрибута. Овај дијалог је приказан на слици 7.



Слика 7 – Дијалог за избор атрибута који жели да се посматра. Одабран је атрибут “GL\_ALPHA\_BIAS”

2. Уклањање атрибута из прозора. Ову акцију је могуће покренути кликом на дугме () које се налази поред назива атрибута или кликом на ставку “Delete” из менија отвореног десним кликом на атрибут који корисник жели да се уклони. Покретање ове акције уклања атрибут из овог прозора.

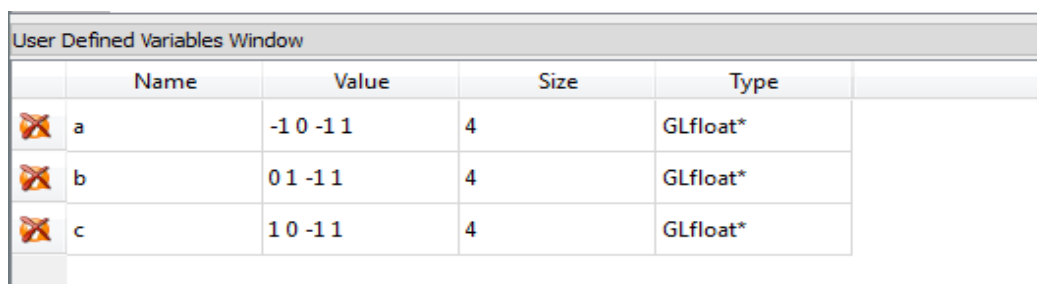
#### Г-2.4. Операције над датотеком




1. Креирање новог програма. Ову акцију је могуће покренути кликом на дугме () или кликом на ставку “Project->New” из главног менија. Покретањем ове акције креира се нови програм. Претходно се корисник упита да ли жели да сачува тренутни програм у датотеци.
2. Отварање програма из датотеке - Ову акцију је могуће покренути кликом на дугме () или кликом на ставку “Project->Open” из главног менија. Покретањем ове акције отвара се дијалог за учитавање програма из датотеке.
3. Снимање програма у датотеку - Ову акцију је могуће покренути кликом на дугме () или кликом на ставку “Project->Save” из главног менија. Покретањем ове акције отвара се

дијалог за снимање програма у датотеку.

## Г-2.5. Прозор за приказ и дефинисање променљивих


На слици 8 је приказан изглед прозора за приказ и дефинисање променљивих.



	Name	Value	Size	Type
	a	-1 0 -1 1	4	GLfloat*
	b	0 1 -1 1	4	GLfloat*
	c	1 0 -1 1	4	GLfloat*


Слика 8 – Приказ прозора за дефинисање променљивих. Дефинисане су променљиве “a”, “b” и “c”.

Акције које су могуће над овим прозором објашњене су у наставку:

1. Дефинисање променљиве. Ову акцију је могуће покренути кликом на дугме () из палете алата, кликом на ставку “Variables->Add Variable” из главног менија, или дуплим кликом на неискоришћен простор овог прозора. Покретање ове акције отвара дијалог где је могуће изабрати атрибут који жели да се посматра. Дијалог нуди кориснику могућност аутокомплетирања уколико корисник жели ручно да унесе име атрибута. Овај дијалог је приказан на слици 9.



Слика 9 – Изглед дијалога за дефинисање променљиве.

2. Уклањање променљиве. Ову акцију је могуће покренути кликом на дугме () које се налази поред назива променљиве или кликом на ставку “Delete” из менија отвореног десним кликом на променљиву коју корисник жели да уклони. Покретање ове акције уклања променљиву. Уколико је променљива адресирана, алат неће дозволити да се променљива уклони.

## Г-2.6. Мени и палете алата

У табели 1 дате су ставке менија, њима одговарајуће дугме из палете алата и акција која им је додељена.

Табела 1 – Ставке менија, одговарајуће дугме из палета алата и акција која им је додељена.

Ставка менија	Дугме	Опис акције	Акција
Project->New		Креирање новог програма	одељак Г-2.4, акција 1
Project->Open		Отварање програма из датотеке	одељак Г-2.4, акција 2
Project->Save		Снимање програма у датотеку	одељак Г-2.4, акција 3
Debug->Continue		Извршавање наредби до наредне тачке заустављања	одељак Г-2.1, акција 3
Debug->Next		Извршавање наредне наредбе	одељак Г-2.1, акција 4
Debug->Next Flush		Извршавање до наредбе за цртање сцене	одељак Г-2.1, акција 5
Debug->Breakpoint		Постављање позиције заустављања програма	одељак Г-2.1, акција 6
Debug->Reset		Заустављање рада коначног аутомата	одељак Г-2.1, акција 7
Calls->Insert Call		Додавање нове наредбе	одељак Г-2.1, акција 8
Calls->Remove Call		Избацивање наредбе	Одељак Г-2.1, акција 9
Calls->Modify Call		Измена наредбе	одељак Г-2.1, акција 10
Calls->Write Call		Ручно писање наредбе	одељак Г-2.1, акција 11
Calls->Move Up		Померање наредбе у смеру на горе	одељак Г-2.1, акција 12
Calls->Move Down		Померање наредбе у смеру на доле	одељак Г-2.1, акција 13
Calls->Region		Формирање региона од секвенце наредби	одељак Г-2.1, акција 14
Calls->Unregion		Расформирање региона у секвенцу наредби	одељак Г-2.1, акција 15
Watch->Add Watch		Додавање атрибута за преглед	одељак Г-2.3, акција 1
Variables->Add Variable		Дефинисање променљиве	Одељак Г-2.5, акција 1

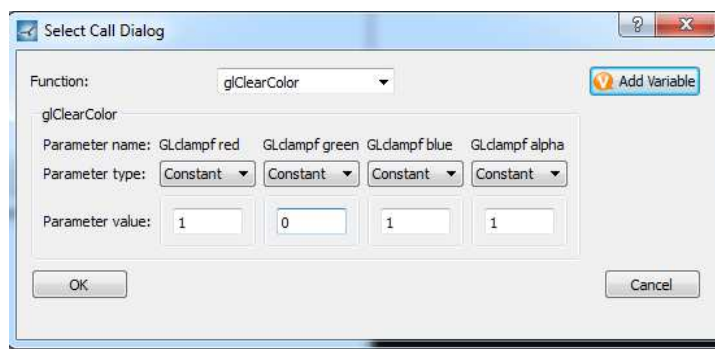
### **Г-3. Примери употребе програма**

У овом делу документа описани су примери употребе програма, кроз четири вежбе.

### Г-3.1. Вежба 1 - Састављање једноставног OpenGL програма

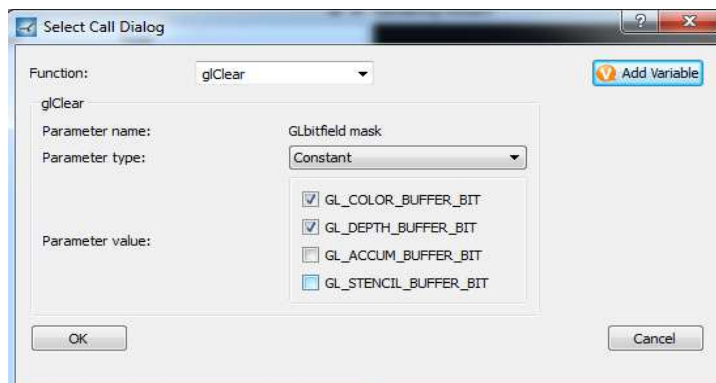
За разлику од конвенционалног писања изворног кода *OpenGL* програма, *SeeGL* нуди алтернативни приступ састављању програма. У наставку текста је дато упутство како корак по корак саставити употребљиви *OpenGL* програм, извршити га и добити резултат цртања сцене. Кораци демонстрирају разне начине састављања програма. У наставку се користи подразумевана конфигурација прозора и палета алата.

1. Уколико корисник није сигуран да већ није изменио подразумевану конфигурацију прозора и палета алата, препоручује се да обрише `SeeGL.layout` датотеку из фолдера где се апликација налази. После брисања рестартовати апликацију.
2. Кликнути на дугме за додавање новог позива у програм (+) и отвориће се дијалог за састављање позива *OpenGL* наредбе.
3. Одабрати `glClearColor` и подесити вредности параметара на 1, 0, 1, 1 (Слика 10). Алат нуди аутоматско комплетирање имена наредбе. Кликнути дугме “OK” и нови позив ће се додати у програм.





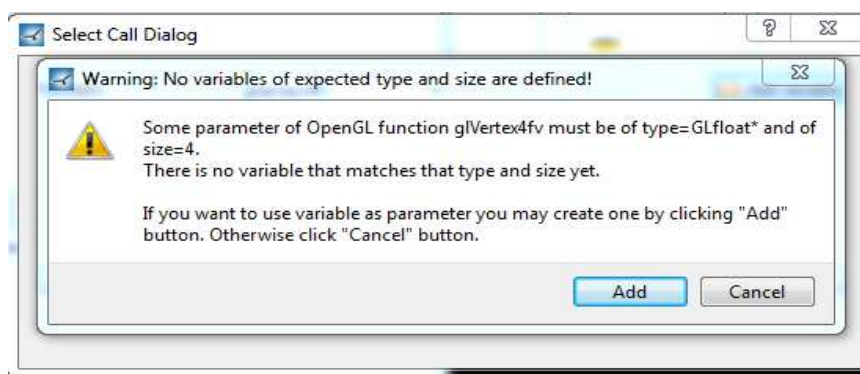
Слика 10 – Састављање позива `glClearColor(1,0,1,1)`

4. Дијалог за састављање наредби је могуће добити и преко менија `Calls->InsertCall`, а и десним кликом на прозор са *OpenGL* програмом, па избором ставке `InsertCall`. Додати на један од ова два начина позив `glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT)`. Обратити пажњу како алат дозвољава само исправне именоване константе за наредбе које их очекују (Слика 11).



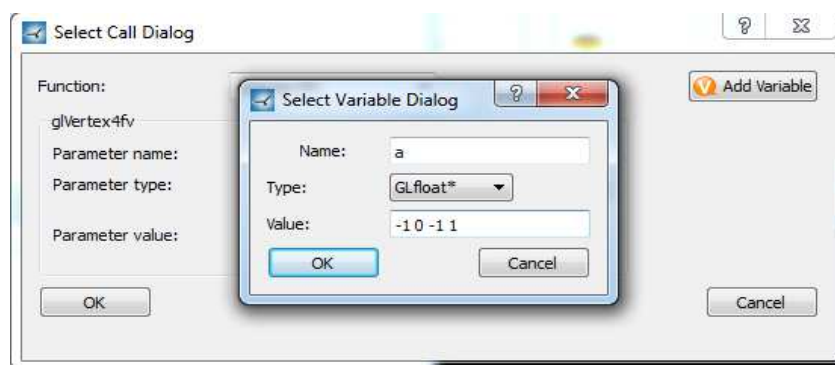
Слика 11 – Састављање позива `glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)`

5. На исти начин додати наредбу `glBegin(GL_TRIANGLES)`. Обратити пажњу да се наредба додаје после одабране наредбе, односно наредбе обојене у плаво. Редослед наредби се једноставно може изменити дугмадима за померање позива горе () , односно доле () .
6. Сада се очекује специфицирање темена троугла. Могуће је користити `glVertex4f` наредбу и уносити вредности темена идентично као што је у претходним корацима урађено. Међутим, да би се објаснио начин употребе променљивих користиће се наредба `glVertex4fv`. Одабрати наредбу `glVertex4fv` и алат ће отворити дијалог-упозорење да корисник није дефинисао променљиву очекиваног типа и очекиване величине (Слика 12).




Слика 12 – Изглед дијалога упозорења да тренутно не постоји ниједна променљива величине 4, типа показивача на `GLfloat`

7. Кликнути на дугме “Add” на дијалогу из претходне тачке и отвориће се дијалог за дефинисање променљиве.
8. На дијалогу за дефинисање променљиве унети име променљиве “a”, тип променљиве: “`GLfloat*`” и вредност поставити на “-1 0 -1 1” као што је приказано на слици 13. *SeeGL* користи карактер “размак” као раздвојник елемената у вектору. Кликнути “OK”.



Слика 13 – Дефинисање променљиве са називом “a”, типа “`GLfloat*`” и са вредношћу “-1 0 -1 1”

9. Корисник је сада поново на дијалогу за одабир позива при чему је променљива “a” већ обележена за адресирање. Кликнути “OK” да би се позив `glVertex4fv(a)` додао у програм.
10. Додати променљиву је могуће на више начина. Дуплим кликом на прозор са променљивама отвара дијалог за дефинисање променљивих. Дугме за дефинисање променљиве () , као и мени ставка `Variables->Add Variable`, такође отварају овај

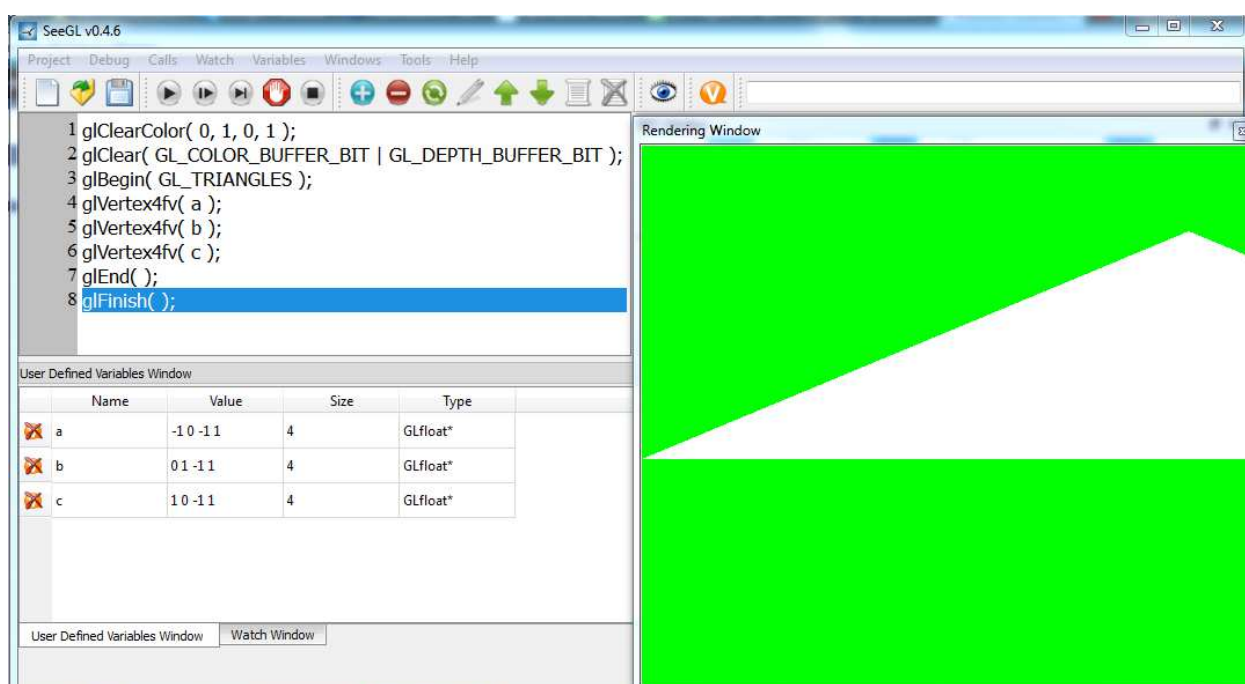
дијалог. Додати променљиве (“b”, “GLfloat\*”, “0 1 -1 1”) и (“c”, “GLfloat\*”, “1 0 -1 1”).

11. Додати још два позива за специфицирање друга два темена: glVertex4fv(b) и glVertex4fv(c)

12. Затим додати наредбе: glEnd() и glFinish().

13. Кликнути на дугме за извршавање наредбе до одабране позиције (▶) и покренуће се коначни аутомат. Показивач текуће наредбе програма ће бити позициониран на прву наредбу, што ће бити означено плавом стрелицом која указује на наредбу која треба следећа да се изврши.

14. Кликнути још једном на иконицу за извршавање програма до одабране позиције и извршиће се програм до краја, али ће оставити коначни аутомат у активном стању. Пошто је извршен позив наредбе glFinish сцена белог троугла (или само његовог дела) се исцртала на зеленој позадини. Резултат би требало да буде сличан као на слици 14.



Слика 14 – Изглед графичког интерфејса на крају вежбе 1

15. Кликнути на дугме за снимање програма (💾) и отвориће се дијалог за снимање у датотеку. Доделити име датотеци и кликнути на дугме за снимање.



16. Затворити програм мени ставком Project->Close или кликом на “x” иконицу.

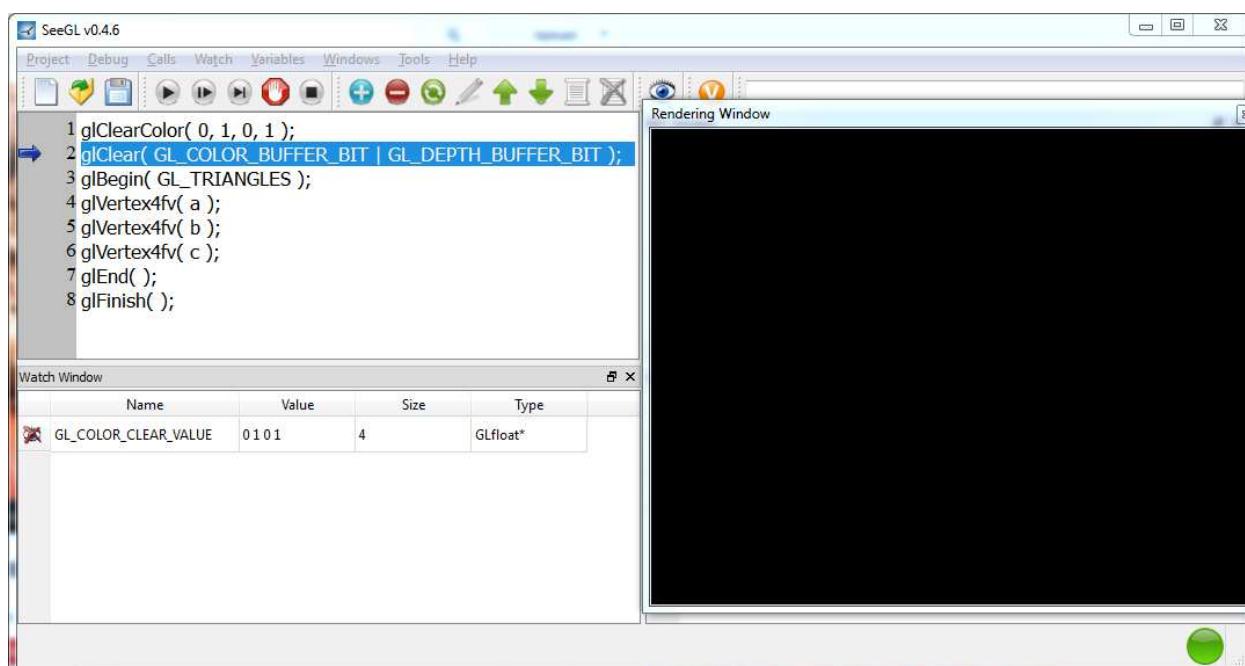
### Г-3.2. Вежба 2 – Анализа програма

У овој вежби су демонстрирани разни начини извршавања програма и читавање вредности атрибута коначног аутомата на означеној позицији између две наредбе програма. Покренути апликацију са подразумеваном конфигурацијом прозора и палета алата.


1. Кликнути на дугме за учитавање програма (📂) и отвориће се дијалог за учитавање из датотеке. Изабрати сачувани програм из вежбе 1.




2. Кликнути на дугме за додавање атрибута за посматрање (  ) и отвориће се једноставан дијалог за одабирање атрибута за посматрање
3. Изабрати атрибут `GL_CLEAR_COLOR_VALUE` и кликнути дугме “OK”. Додати атрибут за посматрање је још могуће користећи мени ставку `Watch->Add Watch` или дуплим кликом миша на прозор са атрибутима за посматрање.
4. Кликнути на дугме за извршавање наредне наредбе (  ) све док се не изврши позив `glClearColor(0,1,0,1)` и показивач следећег позива не буде показивао на позив `glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT)`. Обратите пажњу на прозор атрибута за посматрање и уочити да је вредност атрибута `GL_CLEAR_COLOR_VALUE` ажурирана. Видети слику 15.



Слика 15 – Извршен позив наредбе `glClearColor` је ажурирао вредност атрибута `GL_CLEAR_COLOR_VALUE`



5. Наставити са кликтањем на дугме за извршавање једне наредбе све док се не уђе у режим за задавања темена примитива, што се постиже позивом наредбе `glBegin`. Уочити да је поред вредности атрибута `GL_CLEAR_COLOR_VALUE` назначено да је вредност симулирана. Вредност је морала бити симулирана, јер библиотека `OpenGL` не дозвољава дохватање вредности атрибута за време задавања темена примитива.
6. Поставити позицију заустављања испред позива `glFinish()` на један од два начина: левим кликом миша лево од назива позива `glFinish()` или обележавањем тог позива и кликом на дугме за додавање/избацивање позиције заустављања (  ).
7. Кликнути на дугме за извршавање до одабране позиције заустављања и програм ће извршити све позиве до позиције заустављања. Пошто је позивом наредбе `glEnd()` завршено задавање темена примитивама сада вредност атрибута `GL_CLEAR_COLOR_VALUE` више није симулирана.
8. Кликнути још једном на иконицу за извршавање до одабране позиције заустављања. Отвориће се дијалог да је извршен последњи позив програма, али да *SeeGL* не ресетује коначни аутомат како би омогућио кориснику увид у атрибуте и после извршеног

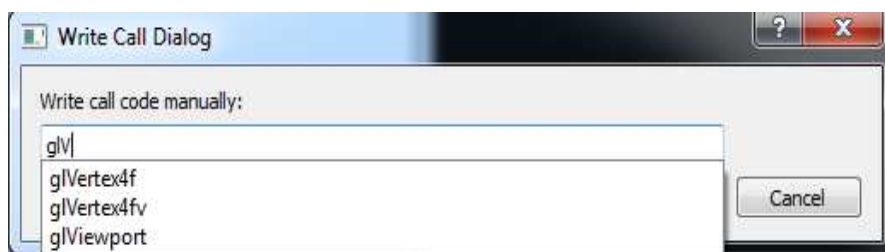
последњег позива. Дијалог затворити кликом на дугме “OK”.

- Искључити коначни аутомат кликом на иконицу за заустављање коначног аутомата (  ). Коначни аутомат би се искључио ако би се програм покренуо из почетка.

### Г-3.3. Вежба 3 – Едитовање OpenGL програма

У овој вежби су објашњене разне могућности едитовања програма са нагласком на све помоћи које алат пружа кориснику приликом едитовања. И ова вежба користи исти програм као и вежба 2, односно онај креиран вежбом 1.

- Дуплим кликом на поље “Value” променљиве “c” обележиће се текст који садржи вредност ове променљиве. Унети нову вредност променљиве “-1 0 -1 1”.
- Обележити позив наредбе glBegin(GL\_TRIANGLES) и кликнути на иконицу за измену позива (  ). Отвориће се исти дијалог као и приликом додавања позива са тим да су овде наредба и вредности параметра већ подешени на позив који би требао да се измени. Изменити вредност GL\_TRIANGLES са вредношћу GL\_QUADS.
- Додати нову променљиву (“d”, “GLfloat\*”, “-1 1 -1 1”)
- Обележити позив glVertex4fv(a) и кликнути на дугме за ручно писање позива (  ). Отвориће се дијалог са већ унетим текстом “glVertex4fv(a)”. Изменити текст позива тако да адресира променљиву “d” уместо променљиве “a”. Кликнути дугме “OK”. Тиме је додат нови позив glVertex4fv(d) одмах испод позива glVertex4fv(a). Дакле, ручно писање позива се може користити као копирање позива (уз могућност измене текста позива копиране наредбе). Приликом ручног писања позива алат нуди аутоматско комплетирање назива наредбе што се може видети на слици 16.



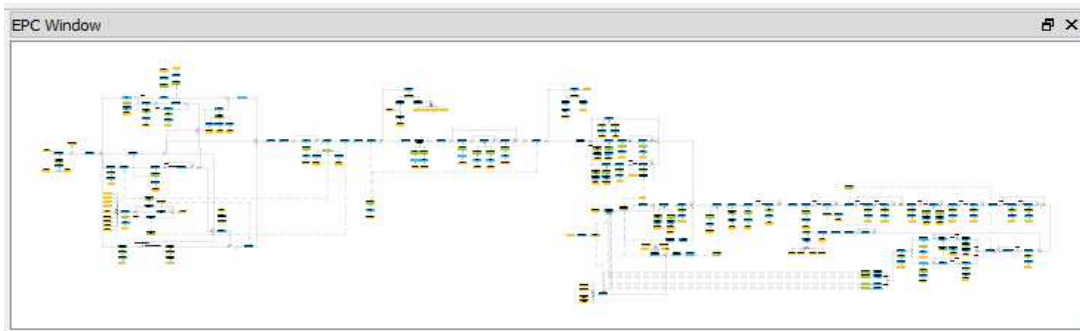
Слика 16 – Аутоматско комплетирање назива наредби приликом ручног писања позива

- Уколико се сада изврши програм резултат сцене би био правоугаоник беле боје на зеленој позадини.

### Г-3.4. Вежба 4 – Интерактивне могућности над приказом ЕРС шеме

У овој вежби су објашњене разне могућности над приказом ЕРС шеме тока обраде података ове библиотеке. И ова вежба користи исти програм као и вежбе 2 и 3, односно онај креиран вежбом 1.

- Кликнути десним кликом на неискоришћен простор приказа ЕРС шеме и кликнути на ставку “View All”. Приказаће се целокупни приказ шеме (слика 17).



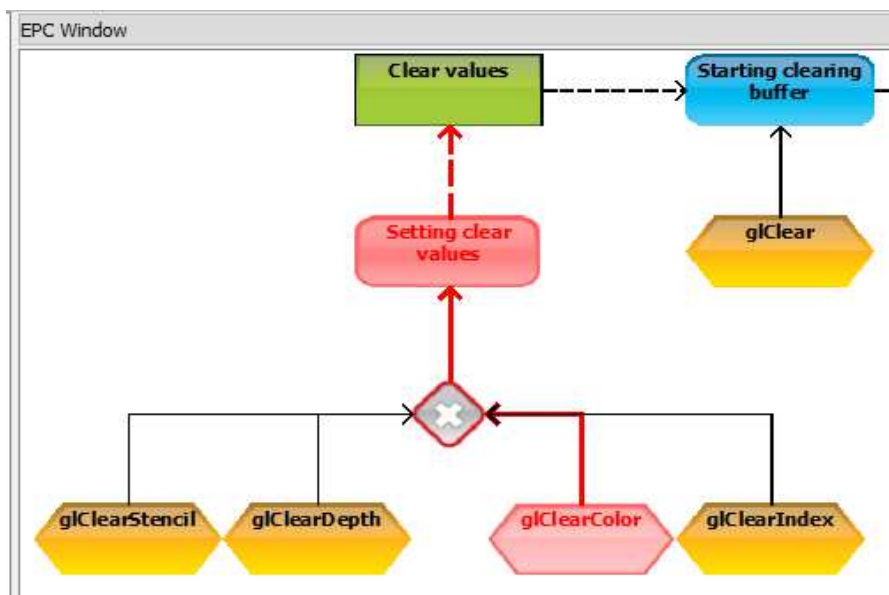
Слика 17 – Целокупни приказ шеме

2. Испробати како ради увећавање и смањивање приказа користећи тачкић на мишу или тастере “+” и “-”.
3. У поље за претраживање (поље десно од палете алата) унети вредност `glClearColor`. Приказ ће се позиционирати тако да је обезбеђен преглед елемента догађаја ове наредбе. Десним кликом на тај елемент отвориће се мени на коме је потребно кликнути на ставку “`Insert glClearColor`”. Отвориће се дијалог за додавање наредбе. Изабрати вредности (0, 1, 0, 1) и кликнути “`Ok`”.
4. Кликнути левим дугметом миша на елемент информационог објекта и отвориће се дијалог са сродним атрибутима приказан на слици 18. Обратите пажњу на ажурирану вредност атрибута `GL_COLOR_CLEAR_VALUE`.

Clear values							
	Name	Value	Size	Type	Create call	Add to watch	View stack
1	<code>GL_COLOR_CLEAR_VALUE</code>	0 1 0 1	4	<code>GLfloat*</code>	<a href="#">Create call</a>	<a href="#">Add to watch</a>	<a href="#">View stack</a>
2	<code>GL_DEPTH_CLEAR_VALUE</code>	1	1	<code>GLdouble*</code>	<a href="#">Create call</a>	<a href="#">Add to watch</a>	<a href="#">View stack</a>
3	<code>GL_STENCIL_CLEAR_VALUE</code>	0	1	<code>GLint*</code>	<a href="#">Create call</a>	<a href="#">Add to watch</a>	<a href="#">View stack</a>
4	<code>GL_INDEX_CLEAR_VALUE</code>	0	1	<code>GLfloat*</code>	<a href="#">Create call</a>	<a href="#">Add to watch</a>	<a href="#">View stack</a>

Слика 18 – Дијалог са сродним атрибутима додељеним елементу “`Clear values`”

5. Кликнути на линк “`Create Call`” атрибута `GL_DEPTH_CLEAR_VALUE` и отвориће се дијалог за додавање наредбе са тим што је већ одабрана наредба која модификује овај атрибут. Кликнути “`OK`” и тиме се у програм додаје наредба `glClearDepth(1)`.
6. Извршити наредбу `glClearColor`. Обратите пажњу на приказ ЕПС шеме који би требао да личи на приказ са слике 19.



Слика 19 – Изглед приказ ЕРС шеме након извршене наредбе glClearColor.

7. Кликнути десним дугметом миша на неискоришћен простор приказа ЕРС шеме и испробати ставку “Dim Inactive” односно “Undim Inactive”.