
madsage
IRan Education
Research
NETwork
(IRERNET)
شبکه آموزشی-پژوهشی مادسیج
با هدف بهبود پیشرفت علمی
و دسترسی راحت به اطلاعات
برای جامعه بزرگ علمی ایران
ایجاد شده است
مادسیج
شبکه آموزشی -پژوهشی ایران
madsg.com
مادسیج

به نام خدا

OpenGL

OpenGLچیست :

OpenGLدقیقا به عنوان یک "رابط نرم افزاری برای سخت افزار گرافیکی" تعریف شده است.OpenGL در ماهیت خود یک کتابخانه مدل سازی و گرافیک سه بعدی میباشد که بسیار سریع و قابل انتقال است. با استفاده از OpenGL شما میتوانید تصاویر سه بعدی زیبا و جذابی طراحی کنید. بزرگترین فایده استفاده از OpenGL اینست که فوق العاده از یک ردیاب نور (ray tracer) سریعتر است OpenGL .از الگوریتمهایی استفاده میکند که توسط شرکت Silicon Graphics توسعه یافته و بهینه شده است .

SGIیک رهبر تائید شده در دنیای گرافیک کامپیوتری و انیمیشن میباشد .

OpenGLیک زبان برنامه نویسی مانند c یا c++ نیست. OpenGL بیشتر شبیه کتابخانه زمان اجرای C می باشدکه یک سری توابع از پیش بسته بندی شده را تدارک دیده. در عمل چیزی به نام برنامه OpenGL وجود ندارد. وقتی ما میگوییم این یک برنامه OpenGL است یعنی در ساختار این برنامه از OpenGL به عنوان API گرافیکی اش استفاده کرده است همانطور که ما از توابعAPI ویندوز استفاده میکنیم تا بتوانیم به فایلها و امکانات شبکه ای و غیره ویندوز دسترسی پیدا کنیم. همین طور هم ما از توابع OpenGL استفاده میکنیم تا بتوانیم گرافیک سه بعدی بلادرنگ طراحی کنیم .

تاریخچه

IRIS GLدر ابتدا یک کتابخانه دوبعدی بود که پیشرفت کرد و به OpenGL تبدیل شد. در حقیقت OpenGL نتیجه تلاشی بود که شرکت SGI برای اصلاح و بهبود IRIS GL کرد.

OpenGLاستاندارد به سازندگان شخصی سخت افزار گرافیکی این اجازه را میدهد که قابلیت های افزودنی خودشان را با عنوان Extensionتهیه کنند که ممکن است بعضی از محدودیت های توابع OpenGL را کم کند یا راحت تر کند و یا اینکه قابلیت های جدیدی را به آن بیفزاید Extension .ها از توابع و ثابت های جدیدی ساخته شده اند که قابلیت های جدیدی را به OpenGL استاندارد می افزایند .

هر سازنده سخت افزار گرافیکی یک اختصار الفبایی مخصوص به خود برای نامگذاری Extension های خودش دارد. برای مثال شرکت NVIDIA از حروف اختصاری NV برای نامگذاری Extension هایی که درست میکنند استفاده میکنند .

OpenGL 2.0توسط شرکت 3 Dlabs ایجاد شد که نگران راکد ماندن و نداشتن یک مدیریت قوی برای OpenGLبود. . این شرکت قابلیت های جدیدی را به OpenGL اضافه کرد که پر اهمییت ترین آنها زبان سایه زنی GLSL بود .

OpenGLچگونه کار میکند :

OpenGLبیشتر از آنکه یک API گرافیکی توصیفی باشد حالت رویه ای دارد. بجای توصیف صحنه و اینکه صحنه چگونه باید ظاهر شود برنامه نویس مراحل لازم را برای دست یافتن به نمایش معین یا یک افکت را تعیین میکند. این مراحل باعث فراخوانی دستورات زیادی از OpenGL میشود. این فرامین برای رسم اشکال ابتدایی گرافیکی مانند خط و نقطه و چندضلعی در صحنه سه بعدی استفاده میشوند .بعلاوه OpenGL نورپردازی و نگاشت بافت و آمیختگی و شفاف نمایی و انیمیشن و بسیاری دیگر از افکت های ویژه سه بعدی و قابلیت های زیاد دیگری را پشتیبانی میکند .

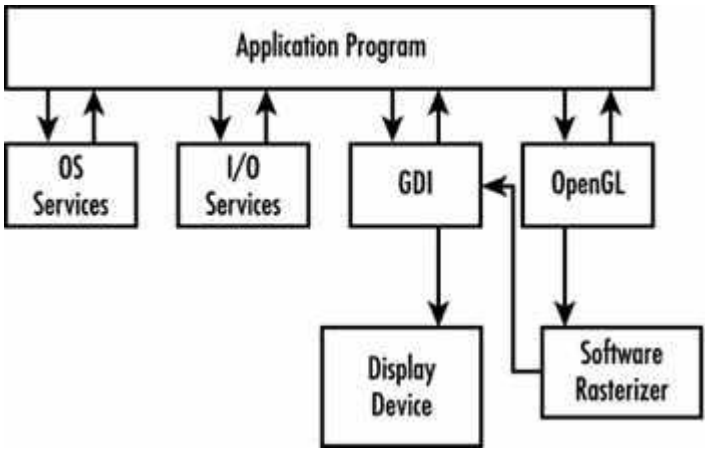
OpenGLشامل هیچ تابعی برای مدیریت پنجره و یا محیط بصری نمیباشد .برنامه نویسان این محیط ها را برای برطرف کردن نیازهای سطح بالایشان ایجاد میکنند و سپس با دقت آنها را با دستورات سطح پایین OpenGL برنامه نویسی میکنند .

پیاده سازی عمومی :

همانطور که قبلا عنوان شد یک پیاده سازی عمومی یک پیاده سازی نرم افزاری میباشد. پیاده سازیهای سخت افزاری برای دستگاههای سخت افزاری ویژه طراحی شده است مانند یک کارت گرافیکی یا یک مولد تصویر .یک پیاده سازی عمومی از لحاظ فنی میتواند بر روی هرکجا اجرا بشود مادامیکه سیستم بتواند تصاویر گرافیکی ساخته شده را نمایش دهد .

تصویر 1-2 مکان نمونه ای را نشان میدهد که OpenGL و یک پیاده سازی عمومی اشغال کرده اند هنگامی که یک برنامه در

حال اجراست. برنامه نمونه توابع زیادی را فراخوانی کرده است. بعضی از توابعی که کاربر تولید کرده و بعضی ها که توسط سیستم عامل مهیا شده اند یا متعلق به کتابخانه زمان اجرای زبان برنامه نویسی هستند. زمانی که برنامه های ویندوز میخواهند که چیزی را بر روی صفحه خروجی رسم کنند معمولا یکی از توابع API ویندوز را که (رابط دستگاه گرافیکی) نامیده میشود صدا میزنند GDI .شامل متد هایی است که به شما اجازه نوشتن متن و ترسیم اشکال دو بعدی ساده و غیره را میدهد



معمولا سازندگان کارت های گرافیکی یک درایور سخت افزاری با رابط های GDI تهیه میکنند که خروجی را بر روی مانیتور رسم کند. یک پیاده سازی نرم افزاری از OpenGL گرافیک های تقاضا شده توسط یک برنامه را میگیرد و از آن گرافیک سه بعدی یک تصویر دو بعدی رنگی ایجاد میکند. سپس این تصویر را به GDI میفرستد تا بر روی مانیتور نمایش دهد. در بقیه سیستم های عامل نیز وضع به همین منوال است اما شما GDI را با سرویس نمایش محلی سیستم عامل خود تعویض میکنید .

OpenGLیک جفت پیاده سازی نرم افزاری مشترک دارد .یکی پیاده سازی نرم افزاری مایکروسافت است که با هر ورژن از ویندوز مانند NT 3.5 و بالاتر و Win95 و 2000 و XP ارایه میشود .

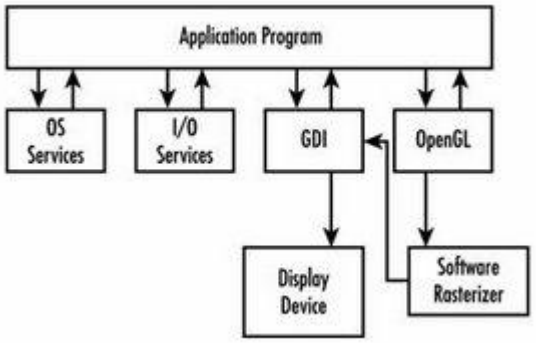
SGIیک پیاده سازی نرم افزاری از OpenGL را برای ویندوز طراحی کرد که پیاده سازی مایکروسافت را از دور خارج میکند .این پیاده سازی دیگر به طور رسمی پشتیبانی نمیشود اما هنوز به مقدار زیادی توسط توسعه دهندگان استفاده میشود. که در انجمن های اوپن سورس از مقبولیت و پشتیبانی خوبی برخوردار است Mesa 3D .یک OpenGL مجوز دار نیست. بنابر این بیش از این که یک پیاده سازی رسمی باشد مانند یک همکار برای OpenGL است .

پیاده سازی OpenGL

واسط OpenGL را به دو روش می توان به کار برد.

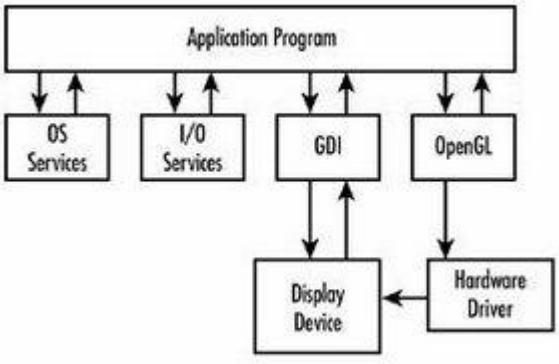
پیاده سازی نرم افزاری

در شکل زیر می توانید موقعیت و نحوه اجرای برنامه هایی که از OpenGL به عنوان سرویس دهنده نرم افزاری استفاده می کنند مشاهده فرمایید. همانطور که می بینید برنامه های کاربردی درخواست های خود را برای OpenGL ارسال می کنند و OpenGL درخواست مربوطه را پردازش کرده و توسط Software Rasterizer یک Image از روی مدل سه بعدی برای GDI ارسال می کند. قسمت GDI در واقع واسطی است که سیستم عامل در اختیار برنامه ها قرار می دهد تا از این طریق با سخت افزار گرافیکی ارتباط بر قرار کنند که در ویندوز به راه انداز سخت افزار گرافیکی GDI می گویند.



پیاده سازی سخت افزاری

در این روش ارتباط بین OpenGL و سخت افزار بدون واسطه و مستقیم می باشد.



به این روش پیاده سازی OpenGL، accelerated implementation می گویند و این نام گذاری به علت بهبود سرعت عملکرد آن می باشد.

همانطور که گفته شد OpenGL یک زبان برنامه نویسی نیست بلکه یک API می باشد که توابع گوناگونی را در اختیار برنامه نویس قرار می دهد. در سیتم عامل ویندوز به طور پیش فرض فایل های glu32.dll و opengl32.dll در پوشه system32 قرار دارند و این توابع را در اختیار برنامه هایی که از فن آوری OpenGL استفاده می کنند قرار می دهد.

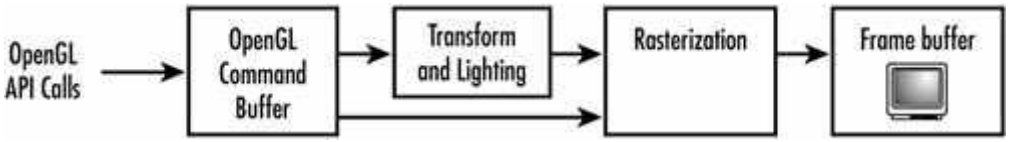
قالب کاری OpenGL در مجموع حدود 300 دستورالعمل را برای ایجاد اشیای گرافیکی، نورپردازی، چرخش و انتقال آن را در اختیار برنامه نویس قرار می دهد. مسئله ای که توجه شما را در اینجا بدان جلب می کنم اینست که OpenGL هیچ گونه

تابعی برای کار با موس یا صفحه کلید در اختیار شما قرار نمی دهد و قابلیت حمل آن ها از همین لحاظ می باشد. زیرا شما می توانید در هر سیستم عاملی از امکانات خود سیستم عامل بهوای تبادل اطلاعات و یا مدیریت رویدادهای محیط گرافیکی و ورودی و خروجی استفاده کنید.

: The Pipeline

کلمه pipeline جهت شرح دادن پروسه ای که میتواند دو مرحله جداگانه یا بیشتر را در بر بگیرد استفاده میشود. تصویرزیر یک pipelineخلاصه شده OpenGL را نشان میدهد .

به عنوان برنامه ای که توابع API مربوط به OpenGL را فراخوانی میکند دستورات در محلی بنام بافر دستور یا Command Bufferذخیره میشود .این بافر بالاخره با اطلاعات راس و تکسچر و غیره پر میشود. وقتی این بافر تا آخرین حد پر شود توسط برنامه یا توسط طراحی درایور دستورات و اطلاعات به مرحله بعدی در پروسه Pipeline پاس داده میشوند



اطلاعات مربوط به رئوس معمولا تغییر شکل یافته هستند. اما برای حالا همین قدر بدانید که "تغییر شکل و نورپردازی" یک مرحله شدیدا ریاضی گونه هستندکه نقاط برای تشریح مختصات هندسی اشیا استفاده میکنند.محاسبات نور پردازی به خوبی بر روی اطلاعات رئوس انجام میشوند تا نشان دهند هر راس با چه شدت رنگی و نوری باید نمایش داده شود.

هنگامی که این مرحله به پایان رسید اطلاعات به بخش بعدی Pipeline یعنی Rasterization خورانده میشود Rasterizer .در عمل یک تصویر رنگی از اطلاعات هندسی و رنگها و اطلاعات تکسچر میسازد. این تصویر سپس به بافر فریم ّ Frame Buffer منتقل میشود. بافر فریم قسمتی از حافظه دستگاه نمایش گرافیکی (کارت گرافیک) میباشد. این بدین معنی است که تصویر در صفحه نمایش داده شده است. در یک سطح بالا این نمودار صحیح میباشد اما در یک سطح پایین تر قسمتهای زیاد دیگری نیز در این پروسه وجود دارد. همچنین استثنائاتی هم وجود دارد. همانطور که در نمودار هم پیداست بعضی از اطلاعات از مرحله T&Lیا همان Transform & Lighting عبور نمیکنند .

در گذشته شتاب دهنده های سخت افزاری OpenGL چیزی جز fast Rasterizer نبودند. آنها تنها بخش Rasterization را شتاب میبخشیدند و پردازشگر سیستم میزبان مرحله T&L را به صورت نرم افزاری و به عنوان بخشی از pipeline انجام میداد. شتاب ده نده های با کیفیت تر) گرانتر) خودشان قسمت T&L را انجام میدادند. به این ترتیب بیشتر مراحل Pipeline در سخت افزار

گرافیکی انجام میشد و گرافیک بیشتری بدست می آمد .امروزه هر کارت ارزان قیمتی مرحله T&L را به صورت سخت افزاری انجام میدهند. خوبی این چیز در اینست که مدلهای با کیفیت بالاتر و تصاویر گرافیکی پیچیده تری در رندر گرافیکی بلادرنگ امکان پذیر میشود.

OpenGLیک API است نه یک زبان برنامه نویسی :

OpenGLیک API است نه یک زبان برنامه نویسی OpenGL .یک رابط برنامه نویسی است. هر زمان که ما میگوییم که این برنامه بر اساس OpenGL است یا OpenGL Based یا این که این یک برنامه OpenGL است منظورمان اینست که این برنامه در یک زبان برنامه نویسی مانند C و یا C++ و یا Java نوشته شده که تعداد یک و یا بیشتر از توابع کتابخانه OpenGL را فراخوانی میکند .

ما نمیگوییم که برنامه تنها از OpenGL برای طراحی استفاده میکند ممکن است که ما از بهترین خصوصیات دو بسته مختلف گرافیکی استفاده کنیم. ممکن است ما از OpenGL برای انجام تعدادی از وظایف ویژه و از GDI برای کارهای دیگری استفاده کنیم. تنها استثنا در این مورد GLSL یعنی همان زبان برنامه نویسی سایه ها در OpenGL میباشد .به عنوان یک API کتابخانه OpenGLاز شیوه صدا زدن تابع در C یا C++ پیروی میکند. آموزشهای ما بر مبنای زبان C میباشد. اما C++ نیز میتواند به سادگی به توابع API مانند C دسترسی پیدا کند. OpenGL در کلیه زبانهای برنامه نویسی از قبیل C و C++ و VB و C# و Delphiو Java و ... قابل استفاده است.

انواع داده در OpenGL

برای اینکه انتقال کدهای نوشته شده در OpenGL به راحتی از یک پلتفرم به پلتفرم دیگر قابل انتقال باشند OpenGL انواع داده مخصوص به خودش را تعریف کرده است. هر محیط برنامه نویسی یا کامپایلری بهر حال قواعد خاصی برای اندازه و نوع حافظه متغیر های مختلف سی "C" دارد. اما با استفاده از انواع داده های OpenGL یعنی OpenGL Data Types شما میتوانید کد خودتان را در برابر این نوع تغییرات عایق کاری کنید .

جدول 1-2 انواع داده OpenGL را لیست کرده است و همچنین انواع داده متناظر آنها در C تحت محیط ویندوز 32 بیتی را نشان داده است .

OpenGL Data Type	Internal Representation	Defined as C Type	C Literal Suffix
GLbyte	8-bit integer	signed char	b
GLshort	16-bit integer	short	s
GLint, GLsizei	32-bit integer	long	l
GLfloat, GLclampf	32-bit floating point	float	f
GLdouble, GLclampd	64-bit floating point	double	d
GLubyte, GLboolean	8-bit unsigned integer	unsigned char	ub
GLushort	16-bit unsigned integer	unsigned short	us
GLuint, GLenum, GLbitfield	32-bit unsigned integer	unsigned long	ui

تمام انواع داده با یک GL شروع میشوند برای ایمکه مشخص باشد از انواع داده OpenGL میباشد و خیلی از آنها از انواع متناظرشان در زبان C منتج شده اند. مانند. … , byte , short , int , float بعضی در ابتدایشان یک حرف u دارند که بیانگر اینست که از نوع بدون علامت "unsigned" میباشد. مانند ubyte که نماینده unsigned byte است .برای بعضی از مصارف روشن ترین و مشخص ترین نام تعیین شده است مانند نوع size که مشخص کننده مقدار طول یا عمق میباشد. بطور مثال GLsizei یکی از توابع OpenGL است که نشان دهنده این است که این تابع یک متغیر از نوع صحیح را به عنوان آرگومان میگیرد .

نقش Clamp یک تذکر جزیی است که نشان دهد مقدار باید در محدوده 0.0 – 1.0 باشد. متغیر های GLboolean بدین منظور استفاده میشوند تا مقادیر صحیح یا غلط را نشان دهند .اشاره گرها و آرایه هاهیچگونه معادل خاصی در OpenGL ندارند. یک آرایه از ده عنصر GLshort بطور ساده به این صورت تعریف میشود :

GLshort shorts[10]; // this way[/align]

و آرایه ای از ده اشاره گر به متغیر های GLdouble بدین صورت تعریف میشود :

[align=left]GLdouble *doubles[10]; //this way[/align] .

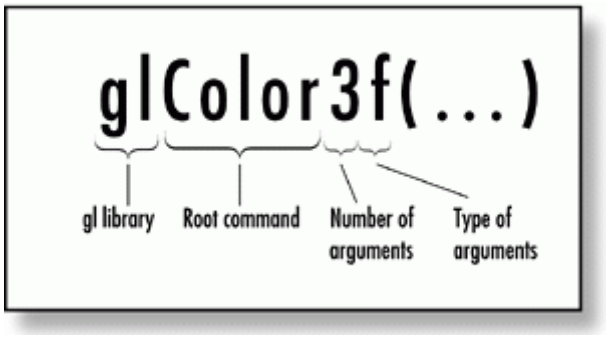
قواعد نامگذاری توابع در OpenGL

بسیاری از توابع OpenGL از یک قرارداد تبعیت میکنند تا به شما بفهمانند این تابع از کدام کتابخانه میباشد و در بعضی مواقع اینکه این توابع چه تعداد و چه نوعی از داده ها را به عنوان آرگومان میپذیرد. تمام توابع ریشه ای دارند که دستور متناظر OpenGLآنها را نشان میدهد. برای مثال تابع glColor3f ریشه اش color است و پیشوند gl نشان دهنده کتابخانه gl میباشد. پسوند 3f نشان دهنده این است که این تابع سه عدد آرگومان از نوع ممیز شناور (floating – point) میگیرد. تمام توابع OpenGL قالب

زیر را دارند:

<Library><Root><Optional><Optional>

تصویر زیر قسمتهای سازنده یک تابع را در openGL نشان میدهد. این تابع نمونه با پسوند 3 fسه عدد آرگومان ممیز شناور میگیرد. نوع دیگر این تابع glColor3i سه عدد صحیح را به عنوان آرگومان میپذیرد. و نوع glColor3d سه عدد از نوع double را به عنوان آرگومان میگیرد و همینطور الی آخر .



این شیوه اضافه کردن تعداد و نوع آرگومانها به انتهای توابع OpenGL به خاطر آوردن لیست آرگومانهای تابع را آسانتر میکند. بعضی از نسخه های تابع glColor چهار آرگومان میگیرد که آرگومان چهارم مربوط به تعیین مولفه آلفا (شفافیت) می باشد. بسیاری از کامپایلر های C/C++ که برای محیط windows ساخته شده اند اینگونه تصور میکنند که هر لیترال ممیز شناور از نوع double است مگر آنکه بطور صریح توسط مکانیسم پسوند به کامپایلر اعلام شود. هنگامی که از لیترال ها (همان حروف اختصاری) برای اعلان آرگومان ممیز شناور استفاده میکنید اگر شما مشخص نکنید که این آرگومانها از نوع float هستند و نه از نوع double کامپایلر در زمان کامپایل یک اخطار میفرستد چون کشف کرده که شما دارید یک متغیر از نوع double را به تابع ارسال میکنید در صورتی که تابع به گونه ای تعریف شده که فقط انواع float را بپذیرد. در نتیجه ممکن است دقت کار از میان برود. موقعی که این اخطار ها به حد زیادی برسد میتواند حتی عمل شناسایی خطاهای نحوی در طول برنامه را مشکل سازد. البته شما میتوانید این اخطار ها را با خاموش کردن قسمت ارسال اخطار کامپایلر خود از بین ببرید اما من شدیدا توصیه میکنم که سعی کنید کدتان را اصلاح کنید تا تمیز و قابل انتقال بماند. پس تک تک اخطارها را با درست کردن شان از بین ببرید. به علاوه ممکن است شما وسوسه بشوید که از توابعی استفاده کنید که آرگومانهایی از نوع ممیز شناور با دقت مضاعف را میپذیرند (double-precision floating-point) به جای float یا. double البته OpenGL بطور ذاتی از نوع float استفاده میکند و در

نهایت انواع دیگر را به نوع ممیز شناور با دقت معمولی تبدیل میکند زیرا هر متغیر از نوع double دو برابر متغیری از نوع floatحافظه مصرف میکند و در مقادیر بالا اینکار OpenGL باعث افزایش سرعت و کارایی برنامه میشود.

OpenGLیک ماشین حالت است :

یک ماشین حالت یک مدل انتزاعی از مجموعه ای از متغیرهای حالت است که میتوانند ارزش های گوناگونی داشته باشند. روشن بودن "on" و یا خاموش بودن "off" و الی آخر. این مقدور نیست که ما هر موقع بخواهیم چیزی را در OpenGL ترسیم کنیم تمام متغیر های حالت را تعیین و مقداردهی نماییم. در عوض OpenGL از یک مدل حالت و یا ماشین حالت استفاده میکند تا رو تمام متغیر های حالت OpenGL را پیگیری نماید. هنگامی که یک متغیر حالت مقدار دهی شد همینطور برقرار میماند تا زمانی که تابع دیگری آن را تغییر دهد. در عمل بسیاری از حالت ها روشن و یا خاموش هستند. برای مثال نورپردازی یا در حالت روشن قرار دارد و یا خاموش است. وقتی نورپردازی در حالت خاموش قرار دارد ترسیمات هندسی ما بدون هیچگونه محاسبات نورپردازی اعمال شده بر روی رنگ اشیای هندسی انجام میگیرد. اما کلیه ترسیمات هندسی پس از روشن کردن حالت نورپردازی به همراه اعمال محاسبات نورپردازی بر روی صفحه ترسیم میشود .

برای روشن کردن این متغیر های حالت شما باید از تابع نمونه زیر کمک بگیرید :

```
void glEnable(GLenum capability);    // this way
```

و برای خاموش کردن از تابع زیر استفاده میکنید :

```
void glDisable(GLenum capability);    // this way
```

اینها شکل کلی این توابع بود. اما در مورد نورپردازی برای مثال شما میتوانید حالت نورپردازی را با استفاده از این حالت روشن (فعال) کنید :

```
glEnable(GL_LIGHTING);    // this way
```

و با کمک این تابع آن را خاموش (غیر فعال) میکنید :

```
glDisable(GL_LIGHTING);    // this way
```

برای اینکه امتحان کنید که یک متغیر حالت فعال است یا نه OpenGL یک مکانیسم راحت را تدارک دیده است

```
Glboolean glIsEnabled(GLenum capability); // this way:
```

البته همانطور که قبلا هم گفتم در نهایت تمام متغیر های حالت یا روشن هستند و یا خاموشند. بسیاری از این توابع میتوانند این متغیر های حالت را ارزش دهی کنند تا وقتی که آنها عوض شوند.شما میتوانید در هر لحظه ای که بخواهید ارزش حالت ها را

چک کنید. یک مجموعه از توابع تحقیق کننده "query functions" به شما اجازه میدهند تا در هر لحظه که بخواهید ارزش هرکدام از متغیر های بولی یا صحیح یا ممیز شناور و یا ممیز شناور با دقت مضاعف را بفهمید. این چهار تابع بدین صورت نمونه سازی شده اند :

void glGetBooleanv(GLenum pname, GLboolean *params); // this way

void glGetDoublev(GLenum pname, GLdouble *params); // this way

void glGetFloatv(GLenum pname, GLfloat *params); // this way

void glGetIntegerv(GLenum pname, GLint *params); // this way

هر تابع یک ارزش واحد و یکتا و یا آرایه ای کامل از ارزشها را برمیگرداند. نگهداری نتایج در آدرسهایی که شما تدارک دیده اید. شما باید ممنون سادگی و قدرت ماشین حالت OpenGL باشید. ذخیره کردن و اعاده (پس دادن یا برگرداندن) حالت ها : OpenGLهمچنین یک مکانیزم راحت و مناسب برای ذخیره کردن محدوده کاملی از ارزشهای حالت و بازگرداندن مجدد آنها دارد. "پشته" یا "stack" یک ساختمان داده ای مناسب است که اجازه میدهد داده ها به پشته وارد شوند (ذخیره کردن) و بعد از پشته بیرون بپرند تا بازیابی شوند. آیتم هایی که قبلا در پشته هل داده شده بودند (ذخیره شده بودند) با دستوری متضاد بازیافت میشوند. ما این را یک ساختمان داده ای بنام "بترتیب عکس ورود "و یا "Last in First Out" مینامیم. دقیقا مثل این میماند که ما بگوییم "هی لطفا این را ذخیره کن" و یا "push it on the stack" و مدت زمانی بعد بگوییم "چیزی که قبلا ذخیره کرده ام را بده" یا "pop it off the stack" در مقالات آتی شما خواهید دید که موضوع پشته "stack" نقش مهمی را در بکارگیری ماتریسها بازی میکند. یک ارزش حالت تنها در OpenGL یا یک محدوده کامل ار ارزشهای حالت مربوطه میتوانند در یک پشته صفت و با کمک دستور زیر قرار بگیرند .

void glPushAttrib(GLbitfield mask); // this way

متقابلا ارزشها با کمک دستور زیر قابل بازیافت هستند :

void glPopAttrib(GLbitfield mask); // this way

خطاها در : OpenGL

واضح است که در هر پروژه ای شما میخواهید که برنامه ای قوی و با تربیت بنویسید که بصورت مودبانه به کاربرانش پاسخ دهد و مقداری هم انعطاف پذیری داشته باشد. برنامه های گرافیکی هم که از OpenGL استفاده میکنند از این قاعده مستثنی نیستند. اگر شما میخواهید که برنامه هایتان بطور روان اجرا شوند شما باید خطاها و رویداد های غیر مترقبه را همیشه مد نظر

قرار دهید .

OpenGLیک مکانیزم مفید و موثر مهیا کرده است که شما با کمک آن بتوانید یک بررسی صحت بر حسب موقعیت "Occasional sanity check"بر روی کدتان قرار دهید. این قابلیت میتواند بسیار مهم باشد .

هنگامی که اتفاقات بدی برای کدهای خوب روی میدهند :

OpenGLبطور ذاتی از مجموعه ای از شش پرچم حمایت میکند که هر پرچم یک نوع خاص از خطا را نشان میدهد. هر زمان که یکی از این خطاها اتفاق بیفتد پرچم متناظر با آن خطا برقرار میشود. برای اینکه بفهمیم آیاهیچ کدام از این پرچم ها برقرارند یا نه این تابع را صدا میزنیم :

glGetError:

Glenum glGetError(void);

تابع glGetError یکی از مقادیر جدول 3-2 را برمیگرداند. کتابخانه GLU سه خطای مخصوص به خودش را تعریف میکند اما این نقشه خطاها دقیقا دو پرچم خطا را ارائه میکنند. اگر بیشتر از یکی از پرچمهابرقرار باشد glGetError هنوز تنها یک پرچم خطا را بر میگرداند. این مقدار پس از فراخوانی glGetError پاک میشود و glGetError دوباره مقدار خطای پرچم بعدی را برمیگرداند و یا مقدار GL_NO_ERROR را برمیگرداند. پس شما نیاز دارید تا تابع glGetError رادر یک حلقه صدا بزنید تا همینطور به چک کردن پرچم های خطا ادامه دهد تا زمانی که مقدار GL_NO_ERROR را برگرداند .

شما همچنین میتوانید از تابع دیگری در کتابخانه GLU استفاده کنید که gluErrorString نام دارد تا رشته ای دریافت کنید که پرچم خطا را شرح میدهد .

const GLubyte* gluErrorString(GLenum errorCode);

این تابع پرچم خطایی را که توسط تابع glGetError برگشت داده شده را به عنوان تنها آرگومانش میگیرد و سپس یک رشته استاتیک را بر میگرداند که خطایی که اتفاق افتاده است را توصیف میکند. پرچم خطای GL_INVALID_ENUM این رشته را باز می گرداند .

Table 2.3. OpenGL Error Codes

Error Code	Description
GL_INVALID_ENUM	The enum argument is out of range.
GL_INVALID_VALUE	The numeric argument is out of range.
GL_INVALID_OPERATION	The operation is illegal in its current state.
GL_STACK_OVERFLOW	The command would cause a stack overflow.
GL_STACK_UNDERFLOW	The command would cause a stack underflow.
GL_OUT_OF_MEMORY	Not enough memory is left to execute the command.
GL_TABLE_TOO_LARGE	The specified table is too large.
GL_NO_ERROR	No error has occurred.

شما میتوانید آسوده خاطر باشید که اگر خطایی بخاطر صدا زدن غیر مجاز توابع OpenGL اتفاق افتاده باشد آن دستور یا تابع نادیده گرفته میشود. تنها استثنا در این مورد توابعی هستند که اشاره گر هایی به حافظه دارند (که میتواند باعث کرش کردن برنامه بشود اگر اشاره گر نامعتبر باشد) .


شبکه آموزشی–پژوهشی مادسیج
با هدف بهبود پیشرفت علمی
و دسترسی راحت به اطلاعات
برای جامعه بزرگ علمی ایران
ایجاد شده است
madsg.com
مادسیج
IRan Education & Research NETwork
(IRERNET)