تکنولوژی ها و ابزارهای ساخت بازی
V 1.0

مقدمه

در سیستم عامل Dos چندین راه برای ترسیم اشکال وجود داشت. ساده ترین راه استفاده از دستورات و توابع گرافیکی زبان برنامه نویسی مورد نظر بود. این روش کندترین روش بود و بنابراین برای عملیاتهای ساده و رسم نمودارها استفاده می شد. روش بعد استفاده از وقفه ها بود. در این روش شما می توانستید با استفاده از وقفه هایی که برای کار گرافیک در نظر گرفته شده بود و همچنین پیاده سازی الگوریتمهای رسم و رنگ آمیزی اشکال توسط خودتان عمل ترسیم کامپیوتری را تا حد ممکن بهینه کنید. با وجود سریعتر بودن این روش باز هم مشکلاتی وجود داشت. اولین مشکل این روش این بود که برای گرافیکهای Real Time باز هم کند بود. دومین اشکال این روش این بود که در صورت پیاده سازی ضعیف الگوریتمها، این روش ممکن بود حتی از روش استفاده از توابع نیز کندتر شود.

روش سوم و سریعترین روش، دسترسی مستقیم به حافظه گرافیکی بود. سیستم عامل همیشه شروع شروع حافظه گرافیک را از آدرس خاصی در نظر می گیرد. حافظه گرافیک در حقیقت حافظه ای بود که هر اطلاعاتی که در آن قرار می گرفت بلافاصله روی صفحه نمایش داده می شد. این روش بسیار سریع بود اما مشکلی که وجود داشت این بود که باز هم سرعت ترسیمات Real Time آنقدرها که انتظار داشتند نبود. بنابراین روش Multi Buffer را ابداع کردند که در این روش در حین نمایش یک فریم در حافظه گرافیک، فریم بعدی در یک بافر ساخته می شد و سپس بافر جدید بلافاصله بعد از اتمام زمان نمایش فریم قبلی درون حافظه گرافیک قرار می گرفت و به همین ترتیب. (ممکن بود تعداد بافرها بیش از یک بافر باشد)

API و 3D API

در ویندوز به دلیل اینکه دسترسی مستقیم به سخت افزار از جمله حافظه امکان پذیر نیست فقط از طریق رابطها (API) می توان به سخت افزار دسترسی داشت. متاسفانه مشکلات کند بودن ترسیم ها دوباره بوجود آمد. ما فقط می توانستیم از طریق GDI ترسیمات گرافیکی را انجام دهیم (GDI مجموعه رابطهای گرافیکی در WinAPI است که برای مستقل بودن عمل ترسیم از سخت افزار ساخته شده اند). به همین دلیل اکثر سازندگان بازی تمایلی برای ساخت بازیهایشان در ویندوز نداشتند و حتی می بینیم که بازیهای DOOM و DUKE با وجود همگانی بودن ویندوز، تحت Dos عرضه شدند.

این مسائل دو شرکت بزرگ Microsoft و Silicon Graphics را به فکر ساخت رابطهایی قویتر و سریعتر تحت ویندوز انداخت که امروزه به آنها 3D API گفته می شود. Microsoft به سرعت دست به کار شد و محصول تلاش او DirectX بود که در نسخه های اولیه چنگی به دل نمی زد اما Silicin Graphics رابطی بسیار قوی و سهل الاستفاده به نام OpenGL را عرضه کرد. این 3D API ها برای ترسیمات 2D و 3D استفاده می شوند. پس از عرضه چندین نسخه از DirectX بالاخره مایکروسافت DirectX 5 را عرضه کرد که جهش زیادی در کیفیت و سرعت نسبت به نسخه های قبلی داشت. با وجود این هنوز آن چیزی نبود که همه انتظارات سازندگان بازی را برآورده کند تا DirectX 6 عرضه شد. دوران امپراطوری DirectX از نسخه 6 به بعد شروع شد. نسخه 8 آن یک جهش بزرگ در زمینه 3D به حساب می آید. DirectX 9 علاوه بر رابطهای قدیمی امکان استفاده از DirectX بصورت Managed Code را در اختیار برنامه نویسان قرار داده است که باز هم جهش بزرگی محسوب می شود (افرادی که هنوز خوشبینانه به Managed Code نگاه نمی کنند بهتر است بدانند که اولین پروژه بازی بصورت Managed Code با نام الکساندر در حال حاضر برای عرضه به بازار آماده شده و مراحل نهایی خود را می گذراند). برای اطلاعات بیشتر در زمینه DirectX می توانید به سایت Microsoft مراجعه کنید.

بر خلاف DirectX که یک 3D API شی گرا محسوب می شود، OpenGL یک 3D API تابع گرا می باشد. همین باعث سادگی استفاده از آن است. با اینحال OpenGL قدرت بسیار خوبی دارد و حتی سازندگان بزرگی مثل ID Software محصولاتش را مبتنی بر OpenGL می سازد (این شامل DOOM 3 هم می شود). OpenGL فقط رابطهای گرافیک را عرضه می کند اما در DirectX علاوه بر رابطهای گرافیک، رابطهایی برای شبکه، صدا، موزیک و فیلم عرضه شده است. به همین دلیل انتخاب اول اکثر سازندگان بازی DirectX است. (آموزش برنامه نویسی با DirectX هم اکنون در سایت Persian-Designers ارائه شده است و ادامه دارد)

از DirectX فقط در سیستم عامل ویندوز می توان استفاده کرد. بنابراین فقط در PC ها و در XboX کاربرد دارد. اما OpenGL بصورت Cross Platform ارائه شده است و در سیستم عاملهای مختلفی امکان استفاده از آن وجود دارد (علت این امر Open Source بودن OpenGL است و هر سازنده سیستم عاملی می تواند نسخه ای از آن را برای سیستم عامل خود بهینه کند). اکثر بازیها و برنامه های گرافیکی در Linux با استفاده از OpenGL نوشته می شوند. در سیستم

عاملهای دیگری مثل MAC هم می توان از OpenGL استفاده کرد. علاوه بر آن بازیهای کنسولی متفاوتی مانند Nintendo و PS و PS2 مبتنی بر OpenGL می باشند. بنابراین بازیهای نوشته شده با OpenGL را می توان (با حداقل تغییر) به طیف وسیعی از سیستم عاملها و منسولها منتقل کرد. اطلاعات بیشتر در مورد OpenGL را می توانید در سایت رسمی آن مطالعه کنید:

www.OpenGL.org

در سیستم عاملهای مختلف انواع دیگری از 3D API ها وجود دارند که کمابیش از آنها نیز استفاده می شود مانند OpenML (Cross Platform) ، SDL (Linux) و ...

حالا اجازه دهید تا با تکنولوژی های دیگر در صنعت بازی سازی و گرافیک آشنا شویم.

GDI+

مجموعه رابطهای گرافیکی در .NET Framework را +GDI می گویند. +GDI امکانات زیادی برای رسم اشکال دارد اما بازهم مشکل کند بودن سرعت ترسیم گرافیکهای Real Time در آن وجود دارد. (بزودی مقاله ای درباره استفاده از +GDI در #C در سایت Persian-Designers ارائه خواهد شد). از +GDI می توان برای ساخت بازیهای دو بعدی که نیاز به ترسیمات پیچیده ندارند و همچنین برای ساخت پازلها استفاده کرد (قبل از اینکه به سراغ 3D API Managed Code بروید، آشنایی با +GDI ضروری است).

Wrapper (لایه ها)

در تقسیم بندی این گروه هنوز ابهاماتی وجود دارد. بعضی از آنها امکاناتی دارند که باعث شده است تا برخی از متخصصان آنها را جزو Engine ها به حساب آورند و بعضی دیگر مشخصاتی دارند که ممکن است جزو 3D API ها به نظر برسند!

Wrapper ها در حقیقت یک لایه جدید روی 3D API ها هستند و برای ساده شدن استفاده از 3D API ها بکار می روند. مثلا به جای استفاده از چندین شی و متدها و خواص آنها برای ساختن چیزی در DirectX می توانید همان کار را با استفاده از یک شی و متدها و خواص جدید آن در Wrapper انجام دهید. یکی از Wrapper های مشهور TrueVision است که سازندگان آن ادعا دارند که TrueVision یک Game Engine است ولی پس از بررسی های مجدد در سایت GameDev این نرم افزار به عنوان یک Warpper شناخته شد! یکی دیگر از Wrapper های مشهور CsGL است که ممکن است در وهله اول 3D API به نظر برسد. CsGL در حقیقت لایه ای برای استفاده از OpenGL در .NET. است.

Game Engine (موتور بازی)

Game Engine به مجموعه ای از روالهای حاوی کد شبیه سازی گفته می شوند که مستقیما رفتار یا منطق بازی و همچنین محیط بازی را مشخص نمی کند. یک موتور بازی حاوی ماژولهایی در رابطه با مدیریت ورودی و خروجی، رندر های 2D و ترسیمات 3D ، صدا، امور مربوط به فیزیک و دینامیک در دنیای مجازی و غیره می باشد.

همانطور که گفته شد یک موتور بازی قسمتهای مختلفی دارد. در حقیقت موتور بازی بسته به نوع بازی به موتورهای اختصاصی تقسیم می شود که مجموع این موتورها، موتور بازی را بوجود می آورند. به تفکیک مهمترین آنها را بررسی می کنیم.

- Graphic Engine : وظیفه آن در موتورهای 3D رندر کردن 3D می باشد. عمدتا در این موتورها از Direct3D و یا OpenGL استفاده می شود. ممکن است عملیات نمایش فیلمها نیز در این موتور انجام شود.
- Sound Engine: تمام کارهای مربوط به صدا و موزیک به عهده این موتور می باشد. عمدتا از DirectSound استفاده می شود.
- Physics Engine: تمام روابط فیزیکی که در دنیای واقعی وجود دارد و باید در بازی نیز در نظر گرفته شود با استفاده از این موتور شبیه سازی می شود مانند نیروی جاذبه.
- Sycology Engine: این جدیدترین نوع از موتورها می باشد که با ظهور بازی های Sims (و بخصوص Sims 2) بوجود آمده اند. در این موتور کلیه روابط انسانی و همچنین اغلب فرضیه های روانشناسی شبیه سازی شده است!
- AI Engine: موتور هوش مصنوعی بازی است که وظیفه آن انجام تمام عملیاتهایی است که تصمیم گیری آن به عهده خود بازی است. مثلا جهت حرکت دشمنان در بازیهای Action و یا عملیات تعقیب و گریز در یک بازی.

ممکن است در ساخت یک بازی انواع دیگری از موتورها نیز وجود داشته باشند که ما در اینجا فقط به مهمترین آنها اشاره کردیم. ضمنا در برنامه نویسی و الگوریتمهای ساخت یک موتور بازی نهایت دقت و سعی خواهد شد تا این الگوریتمها و روشهای پیاده سازی آنها بصورت بهینه باشد تا سرعت کار بالا رود. مقایسه موتورهای بازی نیز بر اساس همین بهینه سازی ها و بالطبع قدرت آنها صورت می گیرد. از جدیدترین و پیشرفته ترین موتورهای بازی می توان به موتور Half Life 2، Doom 3، Unreal 2004 اشاره کرد.

در اینجا نکته مهمی که وجود دارد توجه به تجارت جدید و حرفه ای در این زمینه می باشد. شرکتهای سازنده موتورهای بازی اغلب علاوه بر فروش بازیهای شرکت خود که مبتنی بر آن موتور بازی ساخته شده اند، خود موتورها را نیز به سازندگان دیگر می فروشند و از این طریق هم درآمد فراوانی نصیب شرکتهای ساخت موتور بازی می شود و هم شرکتهای تازه کارتر که هنوز هزینه و نیروی انسانی لازم برای ساخت موتور بازی اختصاصی را ندارند می توانند اقدام به بازیسازی کنند. (به عنوان مثال می توان به بازی Pariah اشاره کرد که بر اساس Unreal Game Engine ساخته شده است و بزودی عرضه خواهد شد).

Game Maker Software (نرم افزارهای بازیساز)

Game Maker ها یا نرم افزارهای بازیساز به نرم افزارهایی گفته می شود که برای ساخت بازیها بکار می روند. این نرم افزارها بصورت 2D و 3D هستند و امکاناتی برای هرچه ساده تر شدن عملیات ساخت یک بازی را فراهم کرده اند و در این نرم افزارها تلاش شده است تا توسعه دهنده بازی تا حد امکان با مفاهیم پیچیده درگیر نشود ضمن اینکه این نرم افزارها اغلب از یک زبان Script مختص به خود نیز برخوردارند که با آنها می توان بصورت حرفه ای تر، از آن نرم افزار استفاده کرد. از رایجترین نرم افزارهای بازیساز 2D که امکانات 3D محدودی را در اختیار شما قرار می دهد می توان GM را نام برد که زبان Script آن GML نام دارد. (آموزش ساخت بازی با GM در حال حاضر در سایت Persian-Designers ارائه شده است). از رایجترین نرم افزارهای بازیساز 3D می توان به 3D Game Studio اشاره کرد که زبان Script آن WDL نام دارد. (آموزش ساخت بازی با این نرم افزار در حال حاضر در سایت Persian-Designers ارائه شده است و ادامه دارد).

اما این نرم افزارها ظاهر قضیه هستند و در پشت صحنه این نرم افزارها از یک Game Engine قوی استفاده شده است که این موتورها نیز مبتنی بر 3D API ها می باشند. بنابراین برای استفاده از این نرم افزارها شما باید ابتدا 3D API مورد نیاز آن را روی سیستم نصب کنید. به عنوان مثال GM 6 از DirectX 8 استفاده می کند. بعضی از این Game Engine ها مختص همان نرم افزار ساخته شده اند بنابراین در زبانهای برنامه نویسی نمی توان از آنها استفاده کرد مانند موتور مورد استفاده در GM . اما بعضی از Game Engine ها علاوه بر خود نرم افزار می توان در زبانهای برنامه نویسی نیز از آنها استفاده کرد. نرم ازفزار 3D Game Studio دارای یک Game Engine به نام A6 می باشد که علاوه بر خود نرم افزار می توان از آن در زبانهای برنامه نویسی مانند C++ نیز استفاده کرد.

Editors (ویرایشگرها)

ویرایشگرها برای ساخت Mod استفاده می شوند. معمولا برای ساخت قسمتهای مختلف در فرمتهای خاص یک بازی از ویرایشگرهایی استفاده می شود که برای بعضی از بازیها به همراه مستندات و ابزارهای دیگر بصورت Game SDK به سازندگان Mod عرضه می شود. ویرایشگرهای مختص هر بازی برای کار با Game Engine همان بازی عرضه می شود. گاهی همه ویرایشگرهای یک بازی در قالب یک ویرایشگر چند کاره عرضه می شود. مثلا می توان از ویرایشگر UnrealEd که یک ویرایشگر چند کاره برای کار با Unreal Game Engine می باشد نام برد. از Doom 3 SDK که شامل ویرایشگرهایی برای قسمتهای مختلف بازی Doom 3 (هوش مصنوعی، بافتها، مراحل و ...) می باشد، می توان برای ساخت Mod مبتنی بر Doom 3 Game Engine استفاده کرد. بعضی از این ویرایشگرها در قالب Plugin برای نرم افزارهای گرافیکی عرضه شده اند.

(در حال حاضر آموزش ساخت Mod بر اساس Farcry Game Engine در سایت Persian-Designers عرضه شده است و ادامه دارد و بزودی آموزش ساخت Mod بر اساس Buttlefield Vietnam Game Engine نیز عرضه خواهد شد)

Game Making Programming Languages (زبانهای برنامه نویسی ساخت بازی)

این زبانهای برنامه نویسی مختص برنامه نویسی بازی ساخته شده اند. این زبانها نیز مانند نرم افزارهای بازیساز بر اساس Game Engine قوی که در پشت صحنه کار می کنند، ساخته شده اند. از معروفترین این زبانها می توان به DarkBasic اشاره کرد. (آموزش DarkBasic در حال حاضر در سایت Persian-Designers عرضه شده است و ادامه دارد). DarkBasic بر اساس نحو زبان Basic ساخته شده است و موتور آن مبتنی بر DirectX می باشد. زبان دیگری به نام DarkGame نیز وجود دارد که بر اساس نحو زبان C++ ساخته شده است که آن هم مبتنی بر DirectX است.

نویسنده: محمد صافدل
نسخه: 1.0
مقاله "تکنولوژی ها و ابزارهای ساخت بازی" قرار است با پیشرفت این تکنولوژی ها و همچنین با مد نظر قرار دادن نظرات اساتید این فن، هر از چند گاهی بروز شود، بنابراین هرگونه سوال، انتقاد، پیشنهاد، اصلاحیه و یا نکات جدیدی که در مورد این مقاله به نظرتان می رسد در تاپیک مخصوص این مقاله که در آدرس زیر ساخته شده است مطرح کنید تا در نسخه های جدیدتر این مقاله در نظر گرفته شوند.
http://www.persian-designers.com/forum/viewtopic.php?t=424