# برنامه نویسی بازی را چگونه آغاز کنیم؟

## یکها و صفرها

قبل از اینکه شما بتوانید بازی بنویسید، باید یاد بگیرید که به کامپیوتر بگویید که چه کاری می خواهید انجام دهید. همه کامپیوترها در حقیقت 0 و 1 ها را می فهمند. من می دانم که این موضوع در ابتدا برای یاد گرفتن دشوار است، اجازه دهید موضوع را روشن کنم. یک کامپیوتر با جریان الکتریسیته کار می کند، بنابراین کامپیوتر می داند که وقتی جریان الکتریسیته برقرار است به معنای 1 و وقتی جریان الکتریسیته برقرار نیست به معنای 0 است. این بهترین راه برای تصور آن است. حالا اجازه دهید گفته هایم را اصلاح کنم، یک 0 (خاموش) واقعا به معنای نبودن مطلق الکتریسیته نیست، در واقع آن نشاندهنده ولتاژ پایین است و مقدار 1 نشان دهنده ولتاژ بالا، اما ما نیاز به فهمیدن بیشتر این مطلب نداریم.

## ارتباط با کامپیوتر

حالا ممکن است بگویید من آمادگی نوشتن هر چیزی را بصورت 0 و 1 دارم؟ نه. این دلیل بوجود آمدن زبانهای برنامه نویسی بود! از آنجایی که فهمیدن هر چیزی بصورت 0 و 1 بسیار بسیار مشکل است، ما لایه هایی را روی 0 و 1 ها توسعه داده ایم. این لایه ها کارها را بسیار آسانتر می کنند. این لایه ها زبانهای برنامه نویسی ای هستند که ما استفاده می کنیم. وقتی ما از زبانهای برنامه نویسی استفاده می کنیم معمولا چیزهایی مانند این می نویسیم:

"If this variable reaches this number then it's time to perform this function. If that function fails, then we need to close the application"

"اگر مقدار متغیر به عدد مورد نظر رسیده است سپس زمان اجرای این تابع فرا رسیده است. اگر اجرای آن تابع شکست خورد، سپس ما باید برنامه را خاتمه دهیم"

این جمله بسیار خوب و قابل فهم است اما کامپیوتر کدهایی که بخوبی در زبان انگلیسی نوشته شده است را به گروهی از 0 و 1 ها تبدیل می کند. بنابراین برای مثال ذکر شده، اگر من کد زیر را بنویسیم:

```
if( NumberOfBulletsUsed > NumberOfBulletsInClip )
{
    TimeToReload();
    ResetBulletsUsedCount();
}
```

در واقع کامپیوتر آن قطعه کد قابل فهم را به چیزی مانند این تبدیل می کند:

"10010010 10010111 01010001 01010100 10101010 10010101 01011011 00100101 01101010 11010100
11010010 10011010 10110010 10101010 01001010 10010101 00100101 01010101 01010010 01010101
10010110 10100100 10101001 10010010 10100101 01001010 01010101 01011011 00100101 01101010
01001010 11010010 10011010 10110010 10101010 01001010 10010101 00100101 01010101 01010010
01010100 10010110 10100100 10101001 10010010 01010101 01010010 01010101 10010110 10100100
10101001".

حتما می گویید که مشکل است و آن را متوجه نشده اید. خبر خوب اینکه ما هیچ وقت تا این اندازه با چیزهایی مانند این درگیر نمی شویم. البته به این نکته توجه کنید که من حقایق را خیلی ساده نشان داده ام، ما احتمالا با پیچیدگی های بیشتری نسبت به 0 و 1 ها روبرو خواهیم بود. بنابراین ما به راهی برای تبدیل کدهای قابل فهم به کدهای غیر قابل فهم نیاز داریم که کامپیوتر بتواند آنها را بفهمد. چکار کنیم؟ بله، از یک برنامه به نام کمپایلر استفاده می کنیم. شما می توانید کمپایلرهای مجانی را از طریق اینترنت و یا انواع تجاری و حرفه ای آن را تهیه کنید. من پیشنهاد می کنم یکی از کمپایلرهای مایکروسافت را بخرید، زیرا به نظر بهترین و رایجترین کمپایلرها هستند. شما می توانید همه کدهایتان را در یک پنجره کمپایلر بنویسید، دقیقا مثل اینکه گزارشی را در Microsoft Word می نویسید. بعد از اینکه کدنویسی را انجام دادید، به کمپایلر می گویید که شروع به کمپایل کردن کند ( چگونگی انجام این کار بستگی به کمپایلر مورداستفاده شما دارد). بعد از اینکه کمپایلر عمل کمپایل کردن کد شما را با موفقیت به اتمام رساند، یک فایل اجرایی به عنوان چیزی که خواسته اید خواهید داشت. حالا اجازه دهید به شما بگویم، کدنویسی مثل قدم زدن در پارک است. شما را دیوانه می کند، معتاد کننده است، یک چالش است، موهای شما را سفید می کند، عمر شما را کوتاه می کند، و بهتر از همه اینکه، زمانی که شما کدنویسی هر چیزی را شروع می کنید، استدلالها شروع می شوند، شما می توانید هر چیزی را در این دنیا به کد ارتباط دهید!

## کدام زبان را باید استفاده کنم؟

همانطوری که زبانهای حقیقی زیادی برای ارتباط بین انسانها استفاده شده است، انواع مختلفی از زبانها وجود دارند که می توانید از آنها برای ارتباط با کامپیوتر استفاده کنید. زبانهای مختلف امکانات و مزایای مختلفی دارند. رایجترین زبان برنامه نویسی کامپیوتر که برای نوشتن بازیها استفاده می شود زبان C++ است. زبان C++ یک زبان شگفت انگیز است. شما می توانید هر کاری را با آن انجام دهید، و همچنین سخت ترین زبان برای یادگیری نیست! البته در مقابل C++ زبانهای دیگری هستند که یادگیری آنها ساده تر است اما به آن اندازه قوی نیستند. برای مثال: Visual Basic، Delphi و Java. یک زبان جدید که به گروه زبانهای برنامه نویسی به تازگی اضافه شده است، زبان C# می باشد. C# برای یادگیری ساده تر از C++ است و تقریبا سریع است. شما می توانید تقریبا هر برنامه ای را که با C++ نوشته اید در زمان کمتری با C# بنویسید. البته C# مشکلات خاص خودش را دارد. اول از همه اینکه آن یک زبان مستقل از پلتفرم نیست. به این معنی که شما نمی توانید کدهایی را که در ماشینهای ویندوز نوشته اید در ماشینهای MAC استفاده

کنید. با C++ می توانید همان کدها (بدون تغییر و یا با تغییرات جزیی) را در یک ماشین MAC و یا در ماشین مبتنی بر ویندوز استفاده کنید.

شما باید خودتان سعی کنید تصمیم بگیرید که با کدامیک از آنها راحتتر هستید. به این نکته توجه کنید که مفهومی به نام زبانهای سطح بالا و زبانهای سطح پایین وجود دارد، بیشتر از زبانهای سطح بالا مانند C++ و Java استفاده می کنند. یک زبان سطح بالا برای ما قابل فهم تر از یک زبان سطح پایین است. برای مثال به این قطعه کد به عنوان کد یک زبان سطح پایین توجه کنید:

```
labe1:
    add edx, 1
    mov eax, edx
    cmp eax, 10
    jz labe1
```

این کد به زبانی به نام اسمبلی نوشته شده است. اسمبلی یک زبان سطح پایین است و هر دستورالعمل مستقیماً به کد قابل فهم کامپیوتر تبدیل می شود. از آن برای بهینه سازی قوی استفاده می شود. به هر حال اگر ما بخواهیم همان برنامه را دقیقا با C++ بنویسیم مانند کد زیر است:

```
while( variable < 10 ) variable++;
```

می بینید. نسخه C++ بسیار کوتاهتر و قابل فهم است. همه آن چیزی که می گوید این است: تا زمانی که مقدار درون متغیر کمتر از ۱۰ است به متغیر یکی اضافه کنید. دقیقا کد قبل همان کار را با زبان سطح پایین انجام می دهد ولی با وضوح کمتر.

من حدس می زنم شما می توانید برنامه نویسی را به عنوان یک منطق ترتیب اصولی از رویدادها توصیف کنید. به نظر می رسد همه چیز مانند مشابه حقیقی آن کار می کند. اگر شما بخواهید یک مهره شطرنج را دو خانه به جلو حرکت دهید همه کاری که باید انجام دهید تغییر موقعیت Y مهره شطرنج به دوخانه جلوتر است. درست است؟ ... اشتباه است! تفاوتهای زیادی بین حرکت دادن یک مهره شطرنج توسط انسان با حرکت دادن مهره شطرنج توسط کامپیوتر وجود دارد. اگر شما خواستید که کامپیوتر یک مهره شطرنج را دو خانه به جلو حرکت دهد باید کارهایی مانند زیر انجام دهید:

- بدست آوردن موقعیت فعلی مهره شطرنج
- بررسی کنید که مهره شطرنج دیگری راه شما را نبسته باشد
- اگر مهره دیگری راه شما را بسته است، توقف کنید
- در غیر این صورت، اگر یک مهره شطرنج در نقطه ای که می خواهید به آن حرکت کنید قرار داشت، آن مهره را حذف کنید.
- در غیر این صورت، اگر همه چیز مهیا بود، عرض یک خانه از صفحه شطرنج را بدست آورید.
- عرض یک خانه شطرنج را در تعداد خانه هایی که می خواهید مهره شطرنج را حرکت دهید ضرب کنید
- مقدار حاصل را به موقعیت فعلی مهره شطرنج در جهت Y اضافه کنید
- موقعیت جدید مهره شطرنج را ثبت کنید تا مهره های دیگر شطرنج بدانند که چیزی در آنجا قرار دارد

به خاطر داشته باشید که کامپیوترها هیچ چیزی نمی فهمند. همیشه این را در خاطر داشته باشید.

### من حالا می توانم همه گفته های ذکر شده را برای ساخت بازی استفاده کنم؟

جواب شما به سادگی نه می باشد. وقتی یک زبان برنامه نویسی برای استفاده از قدرت CPU در انجام کارها خوب است، در بدست آوردن بهترین خروجی از کارت گرافیک شما و ترسیمات آن خوب نیست، امروزه ساخت بازیهای زیبا بدون داشتن کنترل کامل روی کارتهای گرافیک با تکنولوژی های پیشرفته ممکن نیست و برای بدست آوردن این کنترل شما نیاز به استفاده از چیزی اضافه بر زبان برنامه نویسی هستید. چیزی که شما نیاز دارید یک رابط برنامه نویسی کاربردی (API) است. بطور دقیق یک API گرافیکی مورد نیاز شما است. دو تا از مشهورترین API های گرافیکی مورد استفاده تا به امروز OpenGL و Direct3D هستند. حالا شما برای بازی نیاز به راهی برای گرفتن ورودیها خواهید داشت، بنابراین نیاز به یک API ورودی دارید، DirectInput برای همین منظور است و سپس به موزیک و API برای استفاده از صداها نیاز دارید که DirectMusic، DirectSound، FMOD برای این کار پیشنهاد می شوند. شما باید با این همه "Direct" شگفت زده شده باشید. حتما تعجب می کنید اگر بدانید دوستان ما در Microsoft مجموعه ای از این API ها را دارند. آن یک مجموعه از API ها است که می تواند هر کاری را انجام دهد، این مجموعه API را DirectX می نامند. API های موجود در DirectX عبارتند از: Direct3D (برای گرافیکها)، DirectInput (برای ورودی)، DirectMusic (برای صوت)، DirectSound (برای کنترل بیشتر موزیک)، DirectPlay (برای شبکه). پس باید آن را داشته باشید. هر چیزی که شما

واقعا به آن نیاز دارید در DirectX موجود است. دلیل اینکه شما باید این API های اضافه را یاد بگیرید این است که می توانید با سخت افزار ارتباط برقرار کنید و بنابراین می توانید بهترین خروجی را از آن بگیرید.

بنابراین من پیشنهاد می کنم شما ابتدا C++ را یاد بگیرید. به خاطر داشته باشید که چیزهایی مانند این زمان زیادی را برای یادگیری لازم دارد پس لطفا صبور باشید. وقتی شما از C++ استفاده کردید سپس آنچه را که می خواهید انجام دهید. لطفا تا زمانی که به C++ (یا هر زبان برنامه نویسی دیگری که می خواهید استفاده کنید) مسلط نشده اید، شروع به یادگیری DirectX یا OpenGL نکنید. پس از اینکه زبان برنامه نویسی مورد نظرتان را یاد گرفتید، زمان دانلود یک API و کار کردن با آن رسیده است، من به شما پیشنهاد می کنم کیت توسعه نرم افزار DirectX (DirectX SDK) را تهیه کنید زیرا آن همه چیزهایی را که نیاز دارید بصورت مجانی در اختیار شما قرار می دهد. DX SDK برای دانلود حجیم است ولی ارزشش را دارد. در هر صورت شما نیاز به اینها دارید:

- یک کمپایلر
- یک API

۱- شما می توانید از این آدرس یک کمپایلر بگیرید (هر چند که به شما پیشنهاد می کنم که کمپایلر MS Visual C++ 6.0 یا بالاتر از آن را بخرید).
http://www.bloodshed.net
۲- شما می توانید برای یادگیری C++ از کتابهای انلاین این سایت استفاده کنید (پیشنهاد می کنم از جزوه های سایت خودمان استفاده کنید - مترجم)
http://www.mindview.net/Books/TICPP/ThinkingInCPP2e.html
۳- شما می توانید DirectX SDK را از اینجا دانلود کنید:
http://msdn.microsoft.com/library/default.asp?url=/downloads/list/directx.asp

## حالا می توانیم کار را انجام دهیم؟

البته که هنوز نمی توانیم انجام دهیم، شما راه درازی برای رسیدن به آن در پیش دارید. حالا که C++ را یاد گرفته اید و درک خوبی از برنامه نویسی بصورت عمومی دارید، زمان یادگیری چگونگی استفاده کردن از API ها فرا رسیده است، حالا اگر شما DirectX را انتخاب کرده اید، می توانید به یادگیری چگونگی استفاده از آن با استفاده از بخش "برنامه نویسی بازی" در همین سایت شروع کنید. اما اگر شما به سمت چیز دیگری رفته اید، مانند OpenGL، می توانید با استفاده از سایتهایی مانند http://www.gametutorials.com شروع به یادگیری آن کنید.

اجازه دهید به شما بگویم که این یک سواری مفرح در آغاز راه نیست. آغاز راه جایی است که شما در می یابید که توسعه بازی کار شما است یا از پس آن بر نمی آیید. اگر بعد از یک سال یا بیشتر هنوز تالارهای گفتگو در زمینه توسعه بازی را می خواندید و ۱۰ لیتر قهوه را در یک روز بنوشید تا بتوانید برای رفع مشکل پیش آمده در کدهایتان بیدار بمانید، شما، دوست من، هدف خود را یافته اید.

**نتیجه**
آنها چیزهایی هستند که شما باید برای یاد گرفتن شروع کنید. البته همه اینها زمان زیادی برای انجام دادن خواهند گرفت. بعلاوه، من چیزی درباره هنر در توسعه بازی نگفته ام. این همه چیزهایی است که اگر بخواهید برنامه نویسی را یاد بگیرید نیاز دارید. اگر به هنر و موسیقی بیشتر علاقمند هستید، پیشنهاد می کنم طراحی یا آهنگسازی را شروع کنید. به خاطر بسپارید که فقط با خواندن این مقاله قادر به ساخت یک بازی نیستید. برای ساخت یک بازی ماهها تمرین و مطالعه لازم است. بعضی اوقات سالها، خصوصا برای آنهایی که هنوز شروع نکرده اند. فقط تمرین کنید، تمرین کنید، مطالعه کنید، مطالعه کنید.

موفق باشید.

ترجمه: محمد صافدل
منبع: http://www.DevMaster.net