عنوان: **قسمت دوم -- نگاهی دقیق تر به اولین پروژه VC++ (درک مفهوم فایلهای سرآیند و فضای نام ، ویژگیهای زبان C++ و برخی قوانین برنامه نویسی در C++)**
نویسنده: حمیدرضا
تاریخ: ۱۳۹۱/۱۲/۲۷ ۳:۰
آدرس: www.dotnettips.info
برچسب‌ها: Win32 Project, Win32 Console Application, C

در این قسمت نگاهی دقیق‌تر به فایلهای سرآیند ، فضای نام ، ویژگیهای زبان C++ و برخی قوانین برنامه نویسی C++ خواهیم داشت و همچنین در مورد اولین پروژه توضیحات جامع‌تری ارائه میکنیم .

یک برنامه مجموعه ای از دستورات است که توسط کامپیوتر اجرا میگردد ، برنامه نویسان برای نوشتن این دستورات از زبانهای برنامه نویسی استفاده میکنند ، برخی از این زبانها مسقیما قابل فهم توسط کامپیوتر بوده و برخی نیاز به ترجمه دارند . زبانهای برنامه نویسی را میتوان به سه دسته تقسیم نمود :

1 - زبانهای ماشین
2 - زبانهای اسمبلی
3 - زبانهای سطح بالا

زبانهای ماشین :
زبانی که مستقیما و بدون نیاز به ترجمه قابل فهم توسط کامپیوتر می‌باشد . هر پردازنده یا processor زبان خاص خود را دارد !... در نتیجه تنوع زبان ماشین بستگی به انواع پردازنده‌های موجود دارد و اگر دو کامپیوتر دارای پردازنده‌های یکسان نباشتد ، زبان ماشین آنها با یکدیگر متفاوت و غیر قابل فهم برای دیگری می‌باشد . زبان ماشین وابسته به ماشین یا Machine independent می‌باشد . تمامی دستورات در این زبان توالی از 0 و 1 می‌باشند . برنامه‌های اولیه را با این زبان می‌نوشتند در نتیجه نوشتن برنامه سخت و احتمال خطا داشتن در آن زیاد بود . ار آنجا که نوشتن برنامه به این زبان سخت و فهم برنامه‌های نوشته شده به آن دشوار بود ، برنامه نویسان به فکر استفاده از حروف بجای دستورات زبان ماشین افتادند (پیدایش زبان اسمبلی)

زبانهای اسمبلی :
به زبانی که دستورات زبان ماشین را با نمادهای حرفی بیان میکند، زبان اسمبلی (Assembley Language) می‌گویند . چون این زبان مستقیما قابل فهم برای کامپیوتر نیست باید قبل از اجرا آن را به زبان ماشین ترجمه کرد ، به این مترجم اسمبلر گفته میشود . برنامه‌های نوشته شده به این زبان قابل فهم برای برنامه نویس بود اما از آنجا که به ازای هر دستور زبان ماشین یک دستور زبان اسمبلی داشتیم از حجم برنامه‌ها کاسته نشد ! .. بعلاوه چون زبان اسمبلی همانند زبان ماشین از دستورات پایه ای و سطح پایین استفاده میکرد نوشتن برنامه با این زبان هم سخت و مشکل بود . لذا اهل خرد به فکر ابداع نسلی از زبانهای بهتر بودند (پیدایش زبانهای سطح بالا)

زبانهای سطح بالا :
زبانهای سطح بالا قابل فهم بودند و این امکان را داشتند تا چند دستور زبان ماشین یا اسمبلی را بتوان در قالب یک دستور نوشت **(منظور توابع کتابخانه ای در C/C++)** . پس هم فهم ، هم نوشتن برنامه در این زبانها راحت و هم تعداد خطوط کد کمتر شد . این زبانها به زبانهای برنامه نویسی سطح بالا یا High-Level Programming Language معروفند . البته برنامه نوشته شده در این زبان نیز برای کامپیوتر قابل فهم نبوده و باید به زبان ماشین ترجمه شوند ، این وظیفه بر عهده کامپایلر می‌باشد . اولین زبانهای برنامه نویسی سطح بالا مانند PASCAL ، COBOL ، FORTRAN و **C** می‌باشند . زبان برنامه نویسی **C++** تکامل یافته زبان **C** میباشد .
هر یک از زبانهای برنامه نویسی سطح بالا یک روش برنامه نویسی را پشتیبانی میکند به طور مثال زبان C و PASCAL از روشهای برنامه نویسی ساخت یافته ای و پیمانه ای و زبانهای مانند C++ و JAVA از روش برنامه سازی شی گرا یا Object Oriented Programming یا به اختصار (OOP) استفاده میکنند . زبان **C++** چون زبان C را بطور کامل در بر دارد پس از هر سه روش برنامه نویسی ساخت یافته و پیمانه ای و شی گرا استفاده میکند .

تا اینجا با تاریخچه ای از زبانها و مراحل تکامل آنها آشنا شدیم . حال **ویژگیها و دلایل استفاده از زبان C++** را مرور میکنیم :

زبان C در سال 1972 توسط دنیس ریچی طراحی شد . زبان C تکامل یافته زبان BCPL است که طراح آن مارتین ریچاردز می‌باشد ، زبان BCPL نیز از زبان B مشتق شده است که طراح آن کن تامسون بود . (خداوند روح دنیس ریچی را همچون هوگو چاوز با مسیح

بازگرداند ! ...) .

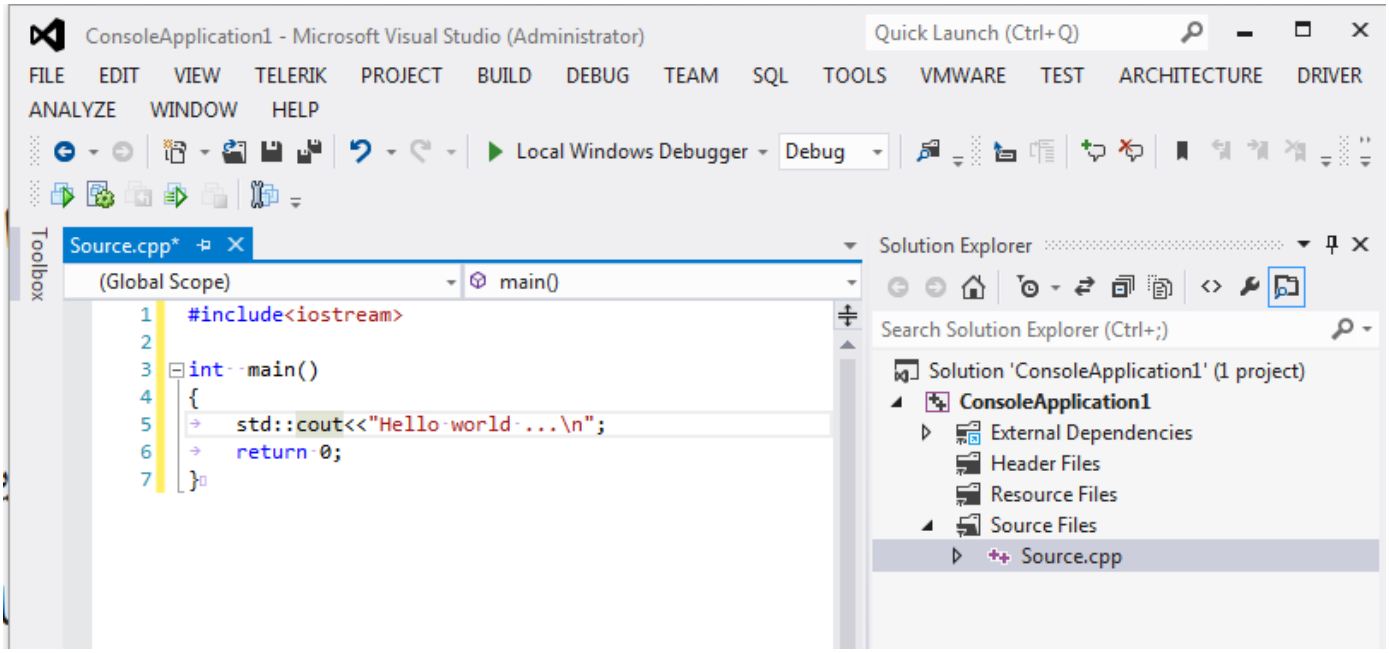
از این زبان برای نوشتن برنامه‌های سیستمی ، همچون سیستم عامل ، کامپایلر ، مفسر ، ویرایشگر ، برنامه‌های مدیریت بانک اطلاعاتی ، اسمبلر استفاده میکنند .

زبان C برای اجرای بسیاری از دستوراتش از توابع کتابخانه ای استفاده میکند و بیشتر خصوصیات وابسته به سخت افزار را به این توابع واگذار میکند لذا نرم افزار تولید شده با این زبان به سخت افزار خاصی بستگی ندارد و با اندکی تغییرات می‌توانیم نرم افزار مورد نظر را روی ماشینهای متفاوت اجرا کنیم ، در نتیجه برنامه نوشته شده با C قابلیت انتقال (Portability) دارند . بعلاوه کاربر میتواند توابع کتابخانه ای خاص خودش را بنویسد و از آنها در برنامه هایش استفاده کند .

برنامه‌های مقصدی که توسط کامپیلرهای C ساخته میشود بسیار فشرده و کم حجم‌تر از برنامه‌های مشابه در سایر زبانهاست ، این امر باعث افزایش سرعت اجرای آنها میشود .

++C که از نسل C است تمام ویژگیهای ذکر شده بالا را دارد ، علاوه بر آن شی گرا نیز می‌باشد . برنامه‌های شی گرا منظم و ساخت یافته اند و قابل آپدیت هستند و به سهولت تغییر و بهبود می‌یابند و قابلیت اطمینان و پایداری بیشتری دارند .

تحلیل اولین پروژه :



در اولین پروژه کد فوق را بکار بردیم ، حال به شرح دستورات آن می‌پردازیم .

```
#include <iostream>
```

دستوراتی که علامت # پیش از آنها قرار میگیرد ، دستورات راهنمای پیش پردازنده هستند . این خط یک دستور پیش پردازنده است که توسط پیش پردازنده و قبل از شروع کامپایل ، پردازش میشود . این کد فایل iostream را به برنامه اضافه میکند . کتابخانه استاندارد ++C به چندین بخش تقسیم شده است و هر بخش فایل سرآیند خود را دارد . دلیل قرار گرفتن این دستور در ابتدای برنامه این است که ، پیش از استفاده از هر تابع و فراخوانی کردن آن در برنامه ، کامپایلر لازم است اطلاعاتی در مورد آن تابع داشته باشد . در خط کد بالا فایل سرآیند iostream استفاده نمودیم زیرا شامل توابع مربوط به I/O (ورودی / خروجی) می‌باشد .

```
int  main()
{
```

```
   return 0;
}
```

دستور فوق بخشی از هر برنامه C++ است ، main تابع اصلی هر برنامه C++ است که شروع برنامه از آنجا آغاز می‌شود . کلمه int در ابتدای این خط ، مشخص میکند که تابع main پس از اجرا و به عنوان مقدار برگشتی (return 0;) یک عدد صحیح باز می‌گرداند .

```
std::cout<<"Hello world ...\n";
```

دستور فوق یک رشته را در خروجی استاندارد که معمولا صفحه نمایش می‌باشد ارسال میکند . std یک فضای نام است . فضای نام محدوده ای است که چند موجودیت در آن تعریف شده است . مثلا موجودیت cout در فضای نام std در فایل سرآیند iostream تعریف شده است .

در زبان C++ هر دستور به ; (سیموکالن) ختم میشود .

نظرات خوانندگان

نویسنده: علیرضا صالحی
تاریخ: ۱۳۹۱/۱۲/۲۷ ۱۳:۱۳

عنوان مطلب صحیح نیست لطفا تغییر دهید،
در فضای برنامه نویسی مراد از VC.NET زبان برنامه نویسی CPP تحت CLI است یعنی با سینتکس CPP و کامپایلر .NET یعنی همان کامپایلری که C#.NET و VB.NET استفاده می‌کنند. یا به عبارتی Managed CPP. در منوی New Project در Visual Studio گزینه CLR باید انتخاب شود.

آموزش هایی که شما ارائه کرده اید مربوط به Native CPP است. برنامه نویسی MFC یا VCL یا Win32 یا.. متفاوت با VC++.NET است.در منوی New Project در Visual Studio گزینه‌های غیر از CLR همگی native هستند.

بهتر است مشابه msdn از واژه Visual C++ استفاده کنید:
Visual C++
.NET Programming in Visual C++

با تشکر

نویسنده: حمیدرضا
تاریخ: ۱۳۹۱/۱۲/۲۷ ۱۳:۴۲

دوست عزیز بنده هم VC++ نوشتم ولی نمیدونم چرا ++ نمایش داده نمیشه .

انشاا... در هر دو مورد کد مدیریت شده و native کد صحبت خواهیم نمود . کد مدیریت شده همانطور که شما فرمودین تحت common language runtime اجرا میشود و برای اجرای برنامه نیاز به نصب دات نت فریمورک روی ماشین مقصد هست ، ولی native کد فقط از توابع کتابخانه ای استفاده میکند و نیازی به نصب .net framework جهت اجرای برنامه بر روی ماشین مقصد ندارد . آموزشهایی که تا کنون داده ام (2 مورد) با توجه به گفته شما مربوط به قسمت native آن می‌باشد .
در ادامه حتما در مورد تفاوت‌های کد مدیریت شده و کد محلی صحبت خواهیم نمود .
تغییر اعمال شد .

نویسنده: علیرضا.م
تاریخ: ۱۳۹۲/۰۱/۰۴ ۲۲:۵۹

سلام
آیا کد native ویژوال سی پلاس پلاس در سایر سیستم عامل‌ها از جمله لینوکس (ابونتو) ران می‌شوند.
اگر نه که به نظرم جاوا خیلی بهتر از سی پلاس پلاس باشد، چون نسخه سازمانی اش توانایی تولید نرم افزارهای native قابل اجرا در تمامی سیستم عامل‌ها رو دارد.

نویسنده: علي
تاریخ: ۱۳۹۲/۰۱/۰۴ ۲۳:۲۹

اگه نه که mono هست و با اون میشه کدهای دات نت رو روی لینوکس هم اجرا کرد.

نویسنده: حمیدرضا
تاریخ: ۱۳۹۲/۰۱/۰۵ ۲:۱۶

با سلام

در جواب : علیرضا.م

در مورد سوال اول برنامه‌های نوشته شده در VC++ ، اینجا رو ببینید .
میتونید از IDEهای دیگه مختص به لینوکس استفاده کنید اینجا رو ببینید .

در مورد سوال دوم باید بگم خود جاوا رو با C نوشتن و برای این منظور که شما فرمودین یا به اصطلاح Cross Platform بودن ، میتونید اینجا و اینجا رو ببینید .

موفق باشید