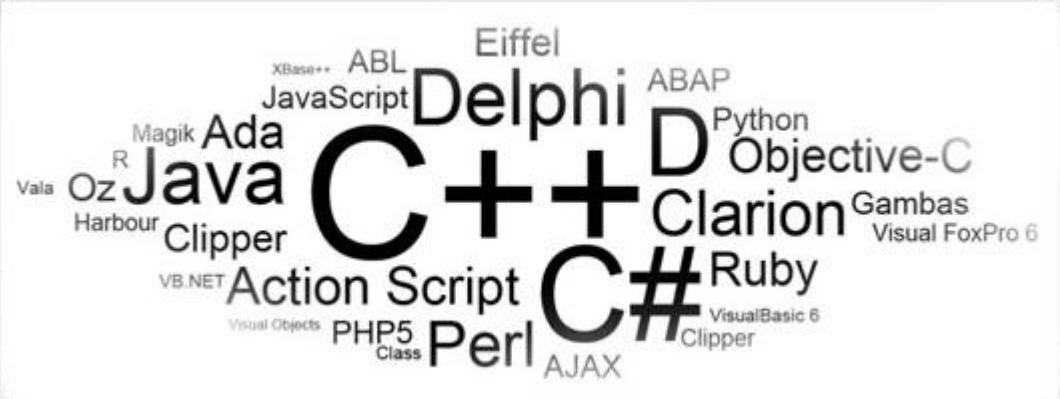
زبان های برنامه نویسی



چکیده مقاله

مسلما یکی از سودمندترین اختراعات بشر تا به امروز کامپیوترها بوده اند. دستگاه هایی که هزاران بار سریعتر از انسان فکر میکنند و سرعت عملشان به طرز غیر قابل باوری بالاست. سرعت و قدرت، امکان انجام خیلی از کارهایی را فراهم کردند که انسان به طور عادی از انجام آن ها عاجز بود. اما یک مشکل وجود داشت: این دستگاه ها به همان اندازه که قوی بودند ، کم هوش هم بودند؛ آن ها به طور عادی هیچ عملی انجام نمی دادند مگر این که به صراحت از آن ها درخواست می شد. اما این درخواست چگونه باید صورت می گرفت؟ باید راهی برای گفتگو با آن ها پیدا می شد. و در این زمان بود که متخصصان تصمیم گرفتند زبان های مخصوصی را بوجود بیاورند تابتوانند با کامپیوترها گفتگو کنند. این زبان های مخصوص به اصطلاح *زبان های برنامه نویسی کامپیوتر* نام گرفتند.به نسل اولیه زبان های برنامه نویسی، زبان های سطح پایین گفته می شد، چرا که به سختی قابل یادگیری و به کارگیری بودند. پیاده سازی اعمالی ابتدایی توسط این زبان ها باعث می شد مدت ها وقت با ارزش برنامه نویسان گرفته شود. اما وضع به همین منوال نماند و با گذشت زمان زبان های جدیدی به وجود آمدند که آموختن آن ها راحت تر بود. طی سه دهه ی اخیر، صنعت کامپیوتر مورد هجوم انواع زبان های برنامه نویسی قرار گرفت.

مقدمه

سیستم های کامپیوتری جدید ، تاثیرات وسیع و رشد یابنده ای بر اکثر فعالیت های بشری داشته و دارند. کامپیوتر امکان داده است تا زمینه های جدیدی از تحقیقات در علوم ایجاد شوند که پیشتر ، به سبب کمبود داده ها و محدودیت در انجام تحلیل ها و محاسبات عددی ، چندان شناخته شده نبودند. کامپیوتر ، پیشرفت های تکنولوژی ، از قبیل سفر به ماه ، را تسهیل کرده و بعنوان وسیله ای برای کنترل فرایندهای صنعتی ، به گستردگی مورد استفاده قرار می گیرد . اکثر سیستم های حسابداری و بانکی ، اینک کامپیوتری شده و در فعالیت هایی مثل مدیریت موجودی و انبار ، پرداخت حقوق حمل و نقل و مراسلات ، از کامپیوتر در حد وسیع استفاده می شود . سازمان های دولتی ، اینک برای ذخیره و بازیابی اطلاعات ، کامپیوتر را بکار می گیرند . در دانشگاه ها برای ذخیره و بازیابی اطلاعات ، امور حسابداری و پرداخت حقوق ، برنامه ریزی دروس و ثبت نام دانشجویان و فعالیت های دیگر از کامپیوتر بهره برداری می شود . بسیاری از سیستم های کتابداری ، اینک کامپیوتری شده اند و در کتابخانه ها ، حتی برای نگاهداری و بازیابی اسناد و مدارک و چکیده های علمی ، از کامپیوتر استفاده می کنند. سخن کوتاه ، کامپیوتر در تمام فعالیت هایی که در آنها پردازش سریع حجم زیادی از اطلاعات ، مورد نیاز باشد، بکار برده می شود .

فصل اول : تاریخچه برخی از زبان های برنامه نویسی

قصد دارم که چند تا از زبان های برنامه نویسی رو معرفی کنم و یه خورده از تاریخچه شون بگم. شاید تکراری باشه برای دوستان برنامه نویس.

زبان برنامه نویسی C

در اوایل دهه ۱۹۷۰ میلادی ، زبان C ، توسط دنیس ریچی و به عنوان زبان برنامه نویسی سیستم ها طراحی شد. این زبان از دو زبان قدیمی تر بنام های BCPL و B حاصل شده است. زبان C تا سال ۱۹۷۸ منحصر به استفاده در لابراتوار کمپانی BELL بود تا این که توسط دو تن بنام های ریچی و کرنیه نسخه نهایی این زبان منتشر شد. به سرعت کامپایلر ها و مفسر های متعددی از C توسعه یافت اما برای جلو گیری از ناسازگاری های ایجاد شده و نیز حفظ قابلیت حمل زبان، تعاریف متحد الشکلی توسط استاندارد ANSI (American national standard institute) موسسه استاندارد ملی آمریکا ارائه گردید.مفسر خود برنامه ای کامپیوتری است که برنامه ی سطح بالا، داده ی ورودی آن و برنامه ی ایجاد شده به زبان ماشین، خروجی آن را تشکیل می دهد. به طور کلی ویژگی های مهم زبان C به اختصار به شرح زیر است:

- زبان سی به طور گسترده ای در دسترس است. مفسر های تجاری آن در بیشتر کامپیوتر های شخصی، مینی کامپیوتر ها و نیز در کامپیوتر های بزرگ قابل استفاده اند.
- سی زبانی است همه منظوره، ساخت یافته سطح بالا (مانند زبان پاسکال و فرتون . ...) و انعطاف پذیر که برخی از خصوصیات زبانهای سطح پایین را نیز که معمولا در اسمبلی یا زبان ماشین موجود است داراست. در عین حال این زبان برای کاربردهای ویژه طراحی نشده و می توان از آن در همه ی زمینه ها، بخصوص به دلیل نزدیکی آن به زبان ماشین در برنامه نویسی سیستم، استفاده کرد. بنابراین سی بین زبان های سطح بالا و سطح پایین قرار دارد و در نتیجه اجازه می دهد که برنامه نویس خصوصیات هر دو گروه زبان را به کار برد. از این رو در بسیاری از کابردهای مهندسی به طور انحصاری زبان سی به کار می برند. (زبان های سطح بالا، دستور العمل هایی شبیه زبان انسان و پردازش فکری او داردند، همچنین یک دستور العمل زبان سطح بالا معادل چند دستور العمل به زبان ماشین است).
- برنامه های نوشته شده به زبان c به طور کلی مستقل از ماشین یا نوع کامپیوتر است و تقریباً تحت کنترل هر سیستم عاملی اجرا می شود.
- کامپایلرهای سی معمولا فشرده و کم حجم اند و برنامه های هدف ایجاد شده با آنها در مقایسه با سایر زبانهای برنامه سازی سطح بالا، خیلی کوچک و کار آمدند.
- برنامه های سی در مقایسه با سایر زبانهای برنامه سازی سطح بالا، به راحتی قابل انتقال اند. دلیل آن این است که سی خیلی از ویژگی های وابسته به نوع کامپیوتر را در توابع کتابخانه ای خود منظور داشته Top of Form
- ست. بنابراین هر نسخه ازسی با مجموعه ای از توابع کتابخانه ای مخصوص به خود همراه است که بر اساس ویژگی های کامپیوتر میزبان مربوط نوشته شده است. این توابع کتابخانه ای تا حدودی استاندارد است و معمولا هر تابع کتابخانه ای در نسخه های متعدد سی به شکل یکسان در دسترس است.
- سی روش برنامه نویسی ماژولار را پشتیبانی می کند. همچنین از نظر عملگر ها نیز زبانی قوی است که عملگرهای گوناگونی برای دستکاری روی داده ها در سطح، بیت داراست.

- به طور کلی جامعیت ، عمومیت، خوانایی، سادگی ، کارآیی، و پیمانه ای بودن که همگی از مشخصات برنامه ای ایده آل اند در زبان C پیاده سازی می شوند.

ویژگی های فوق موجب شده زبان C یکی از قویترین و محبوب ترین زبان های برنامه سازی دنیا مطرح شود.

زبان برنامه نویسی C++

استراس‌تروپ کار بر روی زبان «C با کلاس» را در سال ۱۹۷۹ آغاز کرد.ایده ساخت این زبان جدید در زمان کار بر روی تز دکترای خود به ذهن استراس‌تروپ خطور نمود. او متوجه شد که سیمولا دارای ویژگی‌های مناسب برای ساخت برنامه‌های بسیار بزرگ است اما برای استفاده عملی بسیار کند است اما BCPL با وجود سرعت بسیار زیاد برای ساخت برنامه‌های بزرگ بسیار سطح پایین است. زمانی که استراس‌تروپ کار خود را در آزمایشگاه‌های بل (Bell Labs) آغاز نمود با مشکل تحلیل هسته unix با توجه به محاسبات توزیع شده روبرو شده بود. با یادآوری تجربیات خود در دوران دکترا، او زبان C را با استفاده از ویژگی‌های سیمولا گسترش داد. C به این دلیل انتخاب شد که C یک زبان عمومی، سریع، قابل حمل، و بصورت گسترده در حال استفاده بود. علاوه بر C و سیمولا زبان‌های دیگری مانند ALGOL 68، ADA، CLU، ML نیز بر ساختار این زبان جدید اثر گذاشت. در ابتداویژگی‌های کلاس، کلاس‌های مشتق شده، کنترل نوع قوی، توابع درون خطی، و آرگومان پیش‌فرض از طریق Cfront به C اضافه شد. اولین نسخه تجاری در سال ۱۹۸۵ ارائه شد. در سال ۱۹۸۳ نام زبان از «C با کلاس» به C++ تغییر یافت. ویژگی‌های دیگر شامل توابع مجازی، سربارگزاری عملگر و نام تابع، ارجاعات، ثوابت، کنترل حافظه توسط کاربر بصورت آزاد، کنترل نوع بهتر، و توضیحات یک خطی به صورت BCPL با استفاده از «//» نیز به آن اضافه شد. در سال ۱۹۸۵ اولین نسخه زبان برنامه‌نویسی C++ انتشار یافت و مرجع مهمی برای این زبان فراهم شد در حالی که هیچ استاندارد رسمی وجود نداشت. در سال ۱۹۸۹ ویرایش 20 از زبان C++ ارائه شد. ویژگی‌های جدیدی مانند ارث‌بری چندگانه، کلاس‌های انتزاعی، اعضای ایستای توابع، اعضای ثابت تابع، و اعضای حفاظت شده به آن اضافه شد. در سال ۱۹۹۰ «راهنمای مرجع C++ » منتشر شد. این کار بنیان استانداردهای بعدی شد. آخرین ویژگی‌های اضافه شده شامل موارد زیر بودند: قالب توابع، استثناها، فضاهای نام، تبدیلات جدید، و یک نوع داده منطقی. در حین تکامل C++ کتابخانه استاندارد نیز بوجود آمد. اولین نسخه کتاب استاندارد شامل کتابخانه جریانات I/O بود که جایگزین printf و scanf شد. در ادامه مهم‌ترین ویژگی اضافه شده Standard Template Library بوده‌است.

اهداف به وجود اومدن C++

در کتاب «طراحی و تکامل C++» استراستروپ قوانین مورد استفاده در طراحی C++ را بیان می‌نماید. دانستن این قوانین به فهمیدن نحوه عملکرد C++ و چرایی آن کمک می‌کند. جزئیات بیشتر در کتاب قابل دسترسی است:

- C++ طراحی شده‌است تا یک زبان عمومی با کنترل نوع ایستا و همانند C قابل حمل و پربازده باشد.
- C++ طراحی شده‌است تا مستقیما و بصورت جامع از چندین شیوه برنامه‌نویسی (برنامه‌نویسی ساخت‌یافته، برنامه‌نویسی شی‌گرا، انتزاع داده، و برنامه‌نویسی جنریک)
- C++ طراحی شده‌است تا به برنامه‌نویس امکان انتخاب دهد حتی اگر این انتخاب اشتباه باشد.
- C++ طراحی شده‌است تا حداکثر تطابق با C وجود داشته باشد و یک انتقال راحت از C را ممکن سازد.
- C++ از بکاربردن ویژگی‌های خاص که مانع از عمومی شدن است خودداری می‌نماید.

- C++ از ویژگی‌هایی که بکار برده نمی‌شوند استفاده نمی‌کند.
- C++ طراحی شده‌است تا بدون یک محیط پیچیده عمل نماید.

زبان برنامه نویسی C#

در سال 1999، شرکت سان اجازه استفاده از زبان برنامه نویسی جاوا را در اختیار ماکروسافت قرار داد تا در سیستم عامل خود از آن استفاده کند.جاوا در اصل به هیچ پلت فرم یا سیستم عاملی وابسته نبود، ولی ماکروسافت برخی از مفاد قرار داد را زیر پا گذاشت و قابلیت مستقل از سیستم عامل بودن جاوا را از آن برداشت.شرکت سان پرونده‌ای علیه ماکروسافت درست کرد و ماکروسافت مجبور شد تا زبان شی گرای جدیدی با کامپایل جدید که به C++ شبیه بود را درست کند. در طول ساخت دات نت، کلاس‌های کتابخانه‌ای با زبان و کامپایلر SMC نوشته شدند.در سال 1999 آندرس هلزبرگ گروهی را برای طراحی زبانی جدید تشکیل داد که در آن زمان نامش Cool بود و همانند C بود با خواص شی گرایی. ماکروسافت در نظر داشت اسم این زبان را تا آخر Cool قرار دهد، ولی به دلیل مناسب نبودن برای اهداف تجاری این کار را نکرد. در ارائه و معرفی رسمی .NET در PDC در سال 2000 این زبان به C سی شارپ تغییر نام یافت و کتابخانه کلاس‌ها و runtime در ASP.NET به#C منتقل شدند. مدیر و سرپرست طراحان در ماکروسافت آندرس هلزبرگ بود که تجربه قبلی او در طراحی Framework و زبان‌های برنامه سازیC++ Visual, Turbo Pascal, Delphi , Borland به آسانی در دستورالعمل‌های سی شارپ قابل رویت است و به همان خوبی در هسته CLR.

سی شارپ دارای یک سیستم نوع یکپارچه‌است که به آن CTS می‌گویند.این بدان معناست که تمام انواع،شامل موارد اصلی مانند Integerها، مشتق شده از System.Object هستند.به عنوان مثال، هر نوع یک متد به نام ToString() را به ارث می‌برد.بخاطر کارائی،انواع اولیه (و انواع مقداری) به طور داخلی فضایی برای آنها بر روی پشته در نظر گرفته می‌شود.

زبان برنامه نویسی JAVA

جاوا یک زبان برنامه‌نویسی است که در ابتدا توسط شرکت sun Microsystems ایجاد شده‌است و در سال 1995 به عنوان مولفه اصلی java platform منتشر شد.این زبان قسمت های بسیاری از گرامر خود را از C و C++ گرفته اما دارای مدل شی‌گرایی ساده‌ای است و امکانات سطح پایین کمی دارد. کاربرد جاوا در کامپایل به صورت بایت کد است که قابلیت اجرا روی تمامی ماشین‌های شبیه‌سازی جاوا را داشته باشد صرف نظر از معماری و خصوصیات آن کامپیوتر. اجرای اصلی کامپایلرهای جاوا، ماشین‌های پیاده‌سازی و کتابخانه‌های آن توسط این شرکت از سال 1995 منتشر شد. در may 2007 این شرکت، نرم‌افزار رایگان این زبان را فراهم کرد. دیگران هم کاربردهای دیگری از این زبان را منتشر کردند مثل کامپایلر GNU برای جاوا. Games Gosling پروژه زبان برنامه‌نویسی جاوا را در june 1991 آغاز کرد. این زبان در ابتدا Oak، سپس Green و در آخر هم جاوا نامیده شد. gosling قصد داشت یک ماشین مجازی و یک ماشینی به کار برد که شبیه C و C++ باشد. این شرکت نسخه اول جاوا را تحت عنوان Java1.0 در سال 1995 منتشر ساخت. جستجوگرهای اصلی وب، به هم پیوستند تا به طور مطمئن java applet را بدون صفحات وب اجرا کنند و به این صورت جاوا خیلی زود معروف و محبوب شد. با پیدایش java2، نسخه جدید توانست ترکیب‌های جدیدی را برای نوع‌های مختلف پلت فرم‌ها ایجاد کند. به عنوان مثال J2EE، باهدف کاربرد برای تشکیلات اقتصادی، و نسخه J2ME برای موبایل منتشر شد. در سال 2006 با هدف بازاریابی، این شرکت نسخه جدید J2 را با نام‌های JavaME،JavaEE و JavaSE منتشر کرد. در سال 1997 شرکت سان میکروسیستمز،ISO/IEC JTC1 standards body و Ecma International را به فرمول جاوا تغییر داد. شرکت sun بسیاری از کاربردهای جاوایش را بدون هیچ هزینه‌ای فراهم آورد. شرکت sun با فروش مجوز برای بعضی از کاربردهای خاصش مثل Java Enterprise System درآمدی را بدست آورد. اولین تمایزی که بین SDK و JRE داد شامل فقدان کامپایلر برای JRE و سرفایل ها بود. در 13 نوامبر 2006 شرکت sun نرم‌افزار جاوا را به صورت رایگان و با مجوز عمومی برای همه منتشر کرد.

اهداف اولیه ساخت و طراحی جاوا

- 1. این زبان باید ساده، شی‌گرا و مشهور باشد.
- 2. مطمئن و بدون خطا باشد.
- 3. وابسته به معماری کامپیوتر نبوده و قابل انتقال باشد.
- 4. باید با کارایی بالا اجرا شود.
- 5. باید به صورت پویا و نخ‌کشی‌شده باشد.

زبان برنامه نویسی visual basic

زبان ویژوال بیسیک در واقع حاصل توسعه و ارتقای زبان بیسیک است. بیسیک اولیه حدود سال 1964 کالج دارت موث (Darth Mouth)به وسیله آقایان توماس کورتز (Thomas Kurtz) و جان کمنی (John Kemeny) با هدف گسترش برنامه نویسی بیش دانش آموزان و دانش آموزان و دانشجویان طراحی و ساخته شد.از آن زمان نسخه های متعدد و متفاوتی از آن مانند (QUICK GWBASIC BASIC وTURBOBASIC) ارایه گردید و همواره سعی در افزایش توانمندیهای آن به عنوان یک زبان سطح بالای ساخت یافته شده است . با ظهور سیستم عامل ویندوز 95 و 98 فقدان یک زبان برنامه نویسی آسان و قدرتمند برای استفاده در سیستم عامل های مذکور کاملاً مشهود بود . از این رو مایکروسافت در سال 1991 نسخه اول ویژوال بیسیک را با امکانات یک زبان برنامه نویسی قدرتمند و حرفه ای برای برنامه نویسی در ویندوز ارایه کرد. آخرین نسخه تکامل یافته آن برای استفاده در سیستم عامل جدید مایکروسافت نیز با نام VISUAL BASIC.NET طراحی و ارایه شده است.

یکی از مهم ترین ویژگی های زبان برنامه نویسی ویژوال بیسیک رابط گرافیکی آن است. رابط گرافیکی (GUI) در ویژوال بیسیک یکی از کارآمدترین رابط های گرافیکی در زمینه برنامه نویسی است که به وسیله آن می توان به آسانی برنامه های تحت سیستم عامل ویندوز را ایجاد کرده و حتی قبل از اجرا , شکل ظاهری آن را مشاهده کرد با این که برنامه را به صورت یک مفسر یعنی به صورت خط به خط اجرا نموده و عکس العمل برنامه را بررسی کرد. البته این موارد گوشه ای از ویژگی های متعدد رابط گرافیکی ویژوال بیسیک است.

طراحی سریع برنامه (RAD) یکی دیگر از ویژگی های این زبان است. منظور از طراجی سریع برنامه یا RAD در ویژوال بیسیک این است که طراحی و تولید برنامه ها در ویژوال بیسیک به دلیل وجود ابزارهای مناسب به سرعت انجام می شود . بنابراین هزینه های تولید نرم افزار به طور قابل توجهی کاهش می یابد. ویژگی دیگر زبان برنامه نویسی ویژوال بیسیک ویژگی مدیریت رویداد ها و اتفاقات می باشد ویژوال بیسیک یکی از زبن های برنامه نویسی رویدادگر است . مزیتی که این گونه زبان ها دارند در این است که برنامه نویس می تواند از قبل فرامین لازم را برای وقایع و اتفاقاتی که ممکن است در هنگام اجرای برنامه توسط کاربر رخ دهد سازمان دهی کند.

وجود محیط IDE نیز یکی از ویژگی های مهم این زبان است . محیط IDE به برنامه نویس اجازه می دهد تا برنامه های خود را به سهولت و سرعت , طراحی ,تولید ,خطایابی و اجرا کند . این امکانات به وسیله ابزارهای متعددی که به صورت مجتمع در رابط گرافیکی ویژوال بیسیک قرار داده شده است. قابل دسترسی است.

علاوه بر مواردی که گفته شد دسترسی به برنامه های کاربردی ویندوز به وسیله توابع (API) یکی دیگر از ویژگی این زبان است. توابع API , توابع داخلی ویندوز هستندکه ویژوال بیسیک را قادر می سازد تا با استفاده از فرامین خاصی بتواند به امکانات داخلی موجود در ویندوز دستیابی پیدا کند و برنامه نویس را نیز قادر می سازد تا در صورت نیاز با استفاده از این توابع , برنامه هایی را با توانایی های مورد نظر ایجاد کند.

یکی دیگر از جنبه هایی که تفاوت شگرفی بین ویژوال بیسیک و سایر نسخه های قبلی بیسیک ایجاد می کند امکان استفاده از برنامه نویسی به روش شی گراست . این ویژگی سبب می شود تا ویژوال بیسیک بتواند توقعات برنامه نویس در رابطه با تعریف و به کارگیری اشیا و کلاس های جدید را که سبب راحت تر شدن برنامه نویسی می گردد,برطرف کند. در برنامه نویسی ساخت یافته ,برنامه ها با استفاده از رویه ها به بخش های مختلف تقسیم می شوند که به صورت مجزا از هم قرار می گیرند . دربرنامه نویسی شی گرا با استفاده از اشیا می توان مجموعه ای از دستورالعمل ها و داده ها را در عنصر واحدی به نام شی قرار داد و در زمان مورد نظر از هر یک از بخش های شی مربوطه استفاده کرد.

ویژگی دیگر که در نحوه کار با یک زبان برنامه نویسی مد نظر قرار می کیرد نحوه کشف , تصحیح و برخورد با اشتباهات و خطاهایی است که در هنگام طراحی یا اجرای برنامه رخ می دهدویژوال بیسیک علاوه بر اینکه امکانات بسیار مناسبی در زمینه کشف خطاهای نوشتاری و منطقی برنامه در اختیار برنامه نویس می گذارد .به وی امکان می دهد با استفاده از فرامین مناسب ,خطاهای غیر قابل پیش بینی را نیز در هنگام اجرا تشخیص داده و نحوه ارایه راه حل مناسب را برای راهنمایی کاربران در اختیار آنان قرار دهد. از آغاز ارایه اولین نگارش ویژوال بیسیک , نسخه های متفاوتی از این زبان ارایه شده است. نگارش آموزشی ویژوال بیسیک که برای مصارف آموزشی ارایه شده است. امکان ایجاد برنامه های اجرایی از نوع exe و dll را به همراه استفاده از کنترل های متعدد فراهم می آورد.

نگارش حرفه ای این زبان علاوه بر ویژگی های نگارش آموزشی ,امکان استفاده از کنترل های مربوط به بانک های اطلاعاتی ,طراحی کنترل های ActiveX و هم چنین به کارگیری ویزاردهای مناسب برای تسهیل امر برنامه نویسی را نیز در اختیار برنامه نویسان قرار می دهد و به عنوان کامل ترین نگارش , نگارش نهایی ارایه شده است که در آن امکان برنامه نویسی در شبکه های محلی و اینترنت همراه با ویژگی های سایر نسخه ها فراهم شده است و دارای توانایی استفاده از زبان SQl و تولید و طراحی برنامه های کاربردی با حجم زیاد نیز می باشد.

این سایت تاریخچه یونیکس (تمامی سیستم عامل های یونیکس-بیس، اعم از یونیکس و لینوکس و مک و مینیکس و ...) رو در آدرس زیر داره:

http://www.levenez.com/unix/history.html
/http://www.levenez.com/unix

و هم تاریخچه ویندوز رو داره:

/http://www.levenez.com/windows

http://www.levenez.com/windows/history.html
منابع:مجله الکتریکی فرزان سافت ،ویکی پدیا،http://www.levenez.com/lang /

زبان برنامه‌نویسی پایتون

پایتون یک زبان برنامه‌نویسی سطح بالا، شیءگرا و تفسیری است که توسط گیدو ون روسوم (Guido van Rossum) در سال ۱۹۹۰ طراحی شد.این زبان از زبان های برنامه نویسی تفسیری بوده و به صورت کامل یک زبان شی‌گرا است که در ویژگی‌ها با زبانهای تفسیری پرل، روبی، اسکیم، اسمال‌تاک و تی‌سی‌ال مشابهت دارد و از مدیریت خودکار حافظه استفاده می‌کند.

پایتون پروژه‌ای بازمتن توسعه یافته است و توسط بنیاد نرم‌افزار پایتون مدیریت می‌گردد. تاریخچه پایتون در یک محیط آموزشی ایجاد و توسعه یافته است . یعنی در کریسمس سال ۱۹۹۸ در موسسه ملی تحقیقات ریاضی و رایانه (CWI) در شهر آمستردام. در آن زمان گیدو یک پژوهشگر در CWI بود و در زمان بیکاری خود بر روی پروژه شخصی خود یعنی پایتون کار می کرد . اولین نسخه عمومی از پایتون در ماه فوریه سال ۱۹۹۱ منتشر شد . برای مدتی نسبتاً طولانی پایتون توسط موسسه ملی تحقیقات و ابتکارات (CNRI) واقع در رستون ایالات متحده امریکا

توسعه می‌یافت . تا اینکه در سال ۲۰۰۰ تیم توسعه دهنده پایتون به آزمایشگاه های پایتون منتقل شدند . نام پایتون از برنامه مورد علاقه سازنده آن یعنی مونتی پایتون که یک برنامه کمدی انگلیس بود گرفته شده است .

ویژگی های شی گرایی

پایتون یک زبان برنامه‌نویسی شی‌گرا است و از ویژگی های پیشرفته‌ایی چون وراثت، چند شکلی، سربار گزاری عملگر و ... پشتیبانی می کند. یک از ویژگی‌های پایتون که لقب چسب را برای پایتون به ارمغان آورده امکان استفاده از کد ها و کلاس‌های نوشته شده در زبانهای دیگری چون سی‌پلاس‌پلاس و جاوا است که در حقیقت کار چسباندن قطعات کد جدا و فقط نوشتن بدنه اصلی به عهده پایتون است .

- رایگان

پایتون یک زبان برنامه‌نویسی رایگان و متن‌باز (open source)هست. می‌توانید متن آن و خود برنامه را به رایگان از اینترنت دریافت یا در توسعه آن همکاری کنید .

- قابلیت حمل

چون پایتون با زبان قابل حمل سی نوشته شده می تواند به صورت مجازی بر روی هر پردازشگری همگردانی و اجرا شود . ماشین مجازی (مفسروبندوز بنویسید و سپس بدون تغییر روی لینوکس یا مکینتاش یا هر سیستم عامل و سخت‌افزار دیگری که پایتون روی آن نصب باشد اجرا کنید . پایتون) متن برنامه را خوانده و همزمان تفسیر کرده و اجرا می کند. پس شما می تونید یک برنامه را در

- قدرتمند

پایتون زبانی چند رگه است که از زبان‌های برنامه‌نویسی تفسیری (برای مثال : تی‌سی‌ال، اسکیم، پرل) و زبان‌های سیستمی (برای مثال: سی‌پلاس‌پلاس، سی و جاوا) مشتق شده.

- درونی سازی و گسترش

این ویژگی یکی از پرکاربرد ترین و قوی ترین ویژگی های پایتون می باشد . شما می توانید قطعه از کد را در زبانی چون سی‌پلاس‌پلاس، سی و جاواپایتون استفاده کنید. و یا می توان از توابع کتابخانهای و کامپوننت هایی چون COM API استفاده کرد . البته نوع این نوع برنامه نویسی (ماژول) با برنامه نویسی معمولی هر زبان متفاوت می باشد . می توان از کد های پایتون در زبانهای دیگر نیز استفاده کرد (درونی سازی) نوشته سپس از آن در برنامه نوشته شده

باسهولت یادگیری و استفاده

بی شک و حداقل از نظر بسیاری از برنامه نویسان پایتون این زبان یکی از آسان ترین زبان ها برای یادگیری و استفاده می باشد و از آن به عنوان یک زبان سریع برنامه نویسی یاد می کنند . این زبان نیازی به کامپایل ندارد و شما مستقیما می توانید پس از نوشتن کد و با یک دستور آن را اجرا کنید . دستورات این زبان بسیار نزدیک به زبان انسان می باشد .

منبع : academist

فصل دوم : زبان Delphi

تاریخچه زبان دلفی

دلفی در واقع یک کامپایلر پاسکال است. دلفی ۶ نسل جدید کامپایلر های پاسکال است که شرکت Borland از زمان ایجاد اولین نسخه پاسکال توسط Andres Hejlsberg در ۱۵ سال پیش به بازار عرضه کرد.

برنامه نویسی به زبان پاسکال در سالیان سال از استواری و ثبات، زیبایی و ظرافت و البته سرعت بالای کامپایل سود برده است. دلفی هم از این قاعده مستثنی نیست. کامپایلر دلفی ترکیبی از بیش از یک دهه تجربه طراحی کامپایلر پاسکال و معماری بهبود یافته کامپایلر های ۳۲ بیتی است. اگرچه قابلیت های کامپایلرها با گذشت زمان پیشرفت قابل توجهی داشته است ولی سرعت آن چندان کاهش نیافته و همچنان از سرعت بالایی برخوردار است. به علاوه استحکام و قدرت کامپایلر دلفی معیاری برای سنجش دیگر کامپایلر هاست.

در اینجا به بررسی تفصیلی روند حرکتی دلفی در هر یک از نسخه های آن می پردازیم و مشخصات مهم آن را بررسی می کنیم.

- سال ۱۹۹۵ - Delphi۱

در زمان استفاده از سیستم عامل DOS برنامه نویسان مجبور بودند از بین زبان پر قدرت ولی کم سرعت Basic و زبان کارآمد ولی پیچیده و نامفهوم Assembly یکی را انتخاب کنند. پاسکال با ارایه یک زبان ساخت یافته و یک کامپایلر سریع و کم نقص این شکاف را پرکرد. برنامه نویسان Windows 3.1 هم با تصمیم گیری مشابهی رو برو شدند. یکی زبان قدرتمند و سنگین C++ و یکی زبان ساده و محدود کننده Visual Basic

ارایه Delphi۱ در این مورد هم راه حل خوبی برای برنامه نویسان بود. دلفی مجموعه متفاوتی برای برنامه نویسی بود . طراحی و توسعه برنامه های کاربردی، ایجاد DLL ها، پایگاههای داده و ... که یک محیط ویژوال وسیع را تشکیل می داد. Delphi۱ اولین ابزار برنامه نویسی ویندوز بود که محیط طراحی ویژوال، کامپایلر بهینه کد برنامه و دسترسی قوی به پایگاههای داده را در یک جا جمع کرد که آن را به یکی از بهترین ابزارهای روش نوین توسعه سریع نرم افزار (Rapid Application Development) تبدیل کرد. این مجموعه قدرتمند باعث شد که در همان زمان بسیاری از برنامه نویسان زبانهای دیگر به Delphi روی بیاورند و این موفقیت بزرگی برای Borland به حساب می آمد. همچنین بسیاری از برنامه نویسان پاسکال دلفی را ابزاری یافتند که توسط آن هم از توانایی و تجربه خود در برنامه نویسی پاسکال استفاده می کردند و هم توانایی کار در ویندوز را به دست آوردند. همچنین زبانی که در آن زمان با نام پاسکال شیبی (ObjectPascal) در دانشگاهها ایجاد شده بود یک زبان بسیار خشک و محدود کننده بود که اصلاً حالت کاربردی پیدا نکرد.

ویژگیهای دلفی مثل طراحی ظاهری حساب شده و کاربر پسند آن باعث شد که زبان پاسکال شیبی عملاً از رده خارج شود. تیم طراحی VB در Microsoft قبل از حضور دلفی هیچ رقیب مهمی برای خود نمی دید. VisualBasic در آن زمان زبانی نا کارآمد ، کم سرعت و کند ذهن بود. Visual Basic۳ در عمل اصلا توانایی رقابت با Delphi۱ را نداشت. در این سال شرکت Borland گرفتار یک سری مشکلات قضایی با شرکت Lotus بود که در نهایت هم متخلف شناخته شد. همچنین درگیری مشابهی هم با Microsoft بر سر تلاش در تغییر دادن فضای نرم افزار های Microsoft پیدا کرد. همچنین Borland مشغول طراحی و فروش طرح Quatro به شرکت Novell و طراحی پایگاه های داده dBase و Paradox بود که با استقبال قابل توجهی مواجه نشد.

در این زمان که Borland مشغول فعالیتهای قضایی و تجاری بود Microsft توانست گوی سبقت را از Borland برباید و قسمت اعظم بازار ابزار های برنامه نویسی تحت Windows را در اختیار بگیرد و سعی می کرد تا این طرز فکر را اشاعه دهد که چون Windows را طراحی کرده صلاحیت و توانایی تهیه بهترین ابزار های برنامه نویسی تحت آن را نیز در دست دارد. در این شرایط Borland با عرضه Delphi و نسخه جدید Borland C++ سعی کرد خدشه ای در فرمانروایی Microsoft وارد کند و سهمی در بازار بزرگ این محصولات داشته باشد.

- سال ۱۹۹۶ - Delphi۲

یک سال بعد Delphi۲ تمام مزایای نسخه قبلی را تحت سیستم های جدید ۳۲ بیتی Windows NT۹۵Windows, ارایه داد. همچنین Delphi۲ با ارایه خصوصیات اضافه و کارکرد های قویتری نسبت به Delphi۱ توانایی های خود را افزایش داد از جمله ارایه کامپایلر ۳۲ بیتی که سرعت بالایی به نرم افزار ها می بخشید، کتابخانه بزرگ و کاملی از اشیای مختلف، شیوه جدید و تکامل یافته ای برای اتصال به پایگاه های داده مختلف، ادیتور پیشرفته، پشتیبانی از OLE ، توانایی وراثت در فرمهای ویژوال و سازگاری با پروژه های ۱۶ بیتی. Delphi۱Delphi ۲.۰ به معیاری برای سنجش و مقایسه همه ابزارهای توسعه نرم افزار در آن زمان تبدیل شد.

در آن زمان با ارایه سیستم ۳۲ بیتی Windows۹۵جهش بزرگی در سیستم عامل Windows رخ داد و Borland بسیار مشتاق بود که Delphi را به بهترین ابزار برنامه نویسی سیستم جدید تبدیل کند. نکته این که در آن زمان به منظور تاثیر در افکار عمومی و تاکید بر قدرت Delphi در سیستم عامل ۳۲ بیتی قرار بود که نرم افزار با نام جدید Delphi۳۲ به بازار عرضه شود ولی در آخرین مراحل به خاطر اینکه نشان دهند این زبان زبانی رشد یافته و تکامل یافته نسخه قبلی یعنی Delphi۱ است نام Delphi۲ را برای آن انتخاب کردند.

Microsoft تلاش کرد که با Visual Basic۴ با Delphi مقابله کند ولی از ابتدا کیفیت پایین آن و ضعف آن در انتقال برنامه های ۱۶ بیتی به سیستم ۳۲ بیتی و بروز اشکالات ساختاری در طراحی آن موجب شکست زودهنگام Visual Basic۴ شد. در این زمان هنوز تعداد زیادی از برنامه نویسان به Visual Basic وفادار بودند. Borland هنچنین روشها و ابزارهای قدرتمندی همچون PowerBuilder برای طراحی نرم افزار های Client/Server ارایه داد ولی Delphi هنوز آن قدر قدرتمند نشده بود که بتواند نرم افزارهایی که جایی در بین توسعه گران پیدا کرده اند را براندازد.

- سال ۱۹۹۷ - Delphi۳

از زمان تهیه و توسعه Delphi۱ تیم توسعه Delphi در فکر گسترش و ایجاد یک زبان قدرتمند جهانی بود. برای Delphi۲ این تیم تمام نیروی خود را صرف اعمال مربوط به انتقال تواناییها و کارکرد ها به سیستم ۳۲ بیتی و همچنین اضافه کردن خصوصیات Client/Server و پایگاه داده کرد. در زمان تهیه Delphi۳ تیم توسعه فرصت لازم برای گسترش مجموعه ابزار موجود را یافت و در این راستا کیفیت و کمیت ابزارهای Delphi بهبود یافت. به علاوه راه حل هایی برای مشکلات عمده و قدیمی برنامه نویسان تحت ویندوز ارایه شد. به ویژه استفاده از برخی فناوری های پیچیده و نا مفهوم مثل COM و ActiveX وتوسعه نرم افزار های تحت Web وکنترل پایگاههای داده چند کاربره). روش نمایش کد برنامه همچنین توانایی کامل کردن خودکار کد (Code Completion) عملیات کد نویسی را راحت تر کرد. ضمن این که همچنان در بیشتر موارد اساس و متدولوژی برنامه نویسی مانند Delphi۱ بود و بر پایبندی به قوانین اصولی Pascal تاکید می شد.

در این زمان رقابت شرکت های تولید کننده ابزار های برنامه نویسی بسیار تنگاتنگ شده بودMicrosoft با ارایه Visual Basic۵ به پیشرفت های خوبی دست یافت ازجمله پشتیبانی قوی از COM و ActiveX و ایجاد برخی خصوصیات و تغییرات کلیدی و اساسی در کامپایلر VBضمن این در همین سال Borland با پشتوانه قوی Delphi و با استفاده از ساختار موفق آن ابزارهای دیگری همچون Forte و Builder BC++ به بازار عرضه کرد.

تیم Delphi در زمان طراحی Delphi۳ چند تن از اعضای کلیدی خود را از دست داد. Andres Hejlsberg معمار اصلی Delphi در اقدام غیر منتظره ای Borland را ترک کرد و تصمیم گرفت به رقیب دیرینه یعنی Microsoft بپیوندد. اما حرکت تیم Delphi متوقف نشد و معاون Hejlsberg که سالها تجربه همکاری با او را داشت توانست رهبری این تیم را به خوبی در دست بگیرد. همچنین مسیول فنی تیم (Paul Gross) هم در اقدام مشابهی به گروه Microsoft ملحق شد. این تغییرات بیشتر به خاطر اختلافات شخصی بین افراد تیم بود و نه به خاطر مسایل حرفه ای.

- سال ۱۹۹۸ – Delphi۴

Delphi۴ بیشتر بر روی راحتتر کردن کار با دلفی متمرکز شد. مرورگر روال ها (Module Explorer) بهبود یافت و مرور و ویرایش Unit ها را راحت تر کرد. کنترل کد و کامل کردن خودکار کلاسها این فرصت را به کاربر داد که فکر و زمان خود را روی ساختار اصلی برنامه بگذارد و در وقت صرفه جویی کند. طراحی رابط کاربر هم کاملاً عوض شد و بهبود یافت و اشکال زدا (Debugger) نیز پیشرفت قابل توجهی داشت . Delphi۴ قابلیتهای برنامه نویسان را در استفاده از تکنولوژیهای چند منظوره خارجی مثل MIDAS، DCOM، MIS و Corba افزایش داد.

در این سال Delphi جایگاه خود را در رقابت با دیگران مستحکم کرده بود و کم کم به سمت دست یابی به سودآوری مالی مورد نظر خود پیش می رفت. در واقع در این زمان بود که حاصل کار سنگین چند ساله تیم نمایان می شد. بعد از سالها آزمایش Delphi شهرت و محبوبیت خاصی پیدا کرد و دیگر برنامه نویسان Delphi توانایی جدا شدن از آن را نداشتند. در این زمان Borland به کار سوّال برانگیزی دست زد و به منظور تبلیغ بیشتر و برتری در جنگ روانی با دیگر شرکتها نام Inprise را برای فعالیتهای تجاری خود برگزید.

ابزار های مربوط به فن آوری Corba را گسترش داد تا راه جدیدی برای سودآوری ایجاد کند. برای موفقیت در این زمینه Corba نیاز به رابط کاربر قدرتمندی داشت که در کنار توانایی های آن کار کردن با آن نیز راحت باشد. دقیقاً همان کاری که در سالهای قبل در مورد COM و برنامه نویسی تحت Web انجام شده بود و به موفقیت دست یافته بود. با این وجود بنا به دلایل مختلفی این گسترش و توسعه Corba هیچ وقت تکامل و موفقیتی که مورد نظر بود را به دست نیاورد و بر خلاف تبلیغات و سرمایه گذاری های انجام شده فن آوری Corba تنها توانست نقش کوچکی در روند رو به جلوی Delphi ایفا کند.

- سال ۱۹۹۹ – Delphi۵

Delphi۵ در برخی زمینه ها پیشرفت های قبلی را ادامه داده است. اولاً مسیری را که Delphi۴ با اضافه کردن ویژگیهای زیادی شروع کرده بود ادامه داد. Delphi۴ باعث شد کارهایی که قبلاً به صرف وقت زیادی احتیاج داشت بسیار سریعتر انجام شود. Delphi به شکل امیدوار کننده ای به برنامه نویس این امکان را می دهد که بیشتر به برنامه ای که میخواهد بنویسد توجه کند و نه به قواعد برنامه نویسی و نوشتن کد های تکراری و خسته کننده. این ویژگیهای سودمند شامل رابط کاربر بهبودیافته و سیستم اشکال زدایی(Debugger) توانمند ، امکانات برنامه نویسی تیمی و ابزار های ترجمه می شود.

ثانیا Delphi۵ خصوصیات جدیدی را در بر می گیرد که توسعه برنامه های تحت وب را واقعاً راحت کرده است. این ویژگیها شامل طراح اشیای مربوط به ASP برای ساختن صفحات (Active Server Page)، اشیایی موسوم به Internet Express برای پشتیبانی از XML و خصوصیات جدید MIDAS که آن را به یک ابزار همه کاره در پایگاه های داده تحت Web تبدیل کرد. در نهایت با صرف وقت ، هزینه و صبر زیاد توانست Delphi۵ قدرتمند را عرضه کند. این فعالیت مدتها به طول انجامید و قبل از عرضه عمومی، Delphi۵ بارها در بازبینی ها و آزمایشهای داخلی قسمتهای مختلف آن تغییر کرد و بهبود یافت.

Delphi۵ در نیمه دوم سال ۱۹۹۹ به بازار عرضه شد و به نفوذ و تسلط بر بازار ادامه داد. در این زمان Visual Basic که کم کم به عضوی تحقیر آمیز برای Microsoft تبدیل می شد هم با پیشرفتهایی توانست در رقابت دوام بیاورد و از صحنه خارج نشود. در اقدام درست و به جایی نام Inprise دوباره به Borland بازگشت. این اقدام از سوی طرفداران و مشتریان قدیمی Borland با استقبال خوبی مواجه شد.

- سال ۲۰۰۱ – Delphi۶

در هنگام تهیه Delphi۶ ساختار Delphi در زمینه های مختلف شکل گرفته بود و به یک تکامل نسبی رسیده بود. این مسیله باعث شد که تیم طراحی بتواند وقت خود را بر روی طرحی که مدتها تنها در حد یک نظریه بود بگذارد و آن را بسیار زودتر از آن که انتظار می رفت عملی کند: گام نهادن به محیط های فراتر

از Windows بیشتر نیروی توسعه گران Delphi در این مدت صرف رهانیدن Delphi از بند Windows شد که این خود در درجه اول مبارزه ای آشکار با سلطه Microsoft بود و ثانیاً راه برنامه نویسان را به سوی فضا های دیگر برنامه نویسی باز کرد. در ابتدا این عمل ریسک بزرگی بود و بیم آن می رفت که جایگاه Delphi در Windows هم به خطر بیفتد ولی در نهایت به نقطه رشد و قوتی بدل شد که Delphi را به یکی از بهترین ابزار برنامه نویسی Multi Platform تبدیل کرد.تکنولوژی CLX روالهای مختلف Delphi را با Kylix عضو جدید خانواده Borland که در فضای Linux کار می کند به اشتراک گذاشت و استفاده از سیستم بایت Java باعث شد که Delphi حتی از قید سخت افزار هم رها شود.

به نظر می رسد که این فعالیتها باعث ثبات Delphi در دنیای برنامه نویسان شود و نگرانی های Borland و برنامه نویسان که همیشه می ترسیدند که مبادا با ضعیف شدن Windows جایگاه خود را از دست بدهند حال به افتخار و آرامش برای آنان و نگرانی برای طرفداران Microsoft تبدیل شده است.

فصل سوم:زبان C

تاریخچه C

برای بررسی تاریخچه زبان C باید به سال 1967 بازگردیم که مارتین ریچاردز زبان BCPL را برای نوشتن نرم افزارهای سیستم عامل و کامپایلر در دانشگاه کمبریج ابداع کرد. سپس در سال 1970 کن تامپسون زبان B را بر مبنای ویژگیهای زبان BCPL نوشت و از آن برای ایجاد اولین نسخه های سیستم عامل Unix در آزمایشگاههای بل استفاده کرد. زبان C در سال 1972 توسط دنیس ریچی از روی زبان B و BCPL در آزمایشگاه بل ساخته شد و ویژگیهای جدیدی همچون نظارت بر نوع داده ها نیز به آن اضافه شد. ریچی از این زبان برای ایجاد سیستم عامل Unix استفاده کرد اما بعدها اکثر سیستم عاملهای دیگر نیز با همین زبان نوشته شدند. این زبان با سرعت بسیاری گسترش یافت و چاپ کتاب "The C Programming Language" در سال 1978 توسط کرنیگان و ریچی باعث رشد روزافزون این زبان در جهان شد.

متاسفانه استفاده گسترده این زبان در انواع کامپیوترها و سخت افزارهای مختلف باعث شد که نسخه های مختلفی از این زبان بوجود آید که با یکدیگر ناسازگار بودند. در سال 1983 انستیتوی ملی استاندارد آمریکا (ANSI) کمیته ای موسوم به X3J11 را را مامور کرد تا یک تعریف فاقد ابهام و مستقل از ماشین را از این زبان تدوین نماید.در سال 1989 این استاندارد تحت عنوان ANSI C به تصویب رسید و سپس در سال 1990، سازمان استانداردهای بین المللی (ISO) نیز این استاندارد را پذیرفت و مستندات مشترک آنها تحت عنوان ANSI/ISO C منتشر گردید.

در سالهای بعد و با ظهور روشهای برنامه نویسی شی گرا نسخه جدیدی از زبان C بنام C++ توسط بیارنه استراوستروپ در اوایل 1980 در آزمایشگاه بل توسعه یافت. در C++ علاوه بر امکانات جدیدی که به زبان C اضافه شده است، خاصیت شی گرایی را نیز به آن اضافه کرده است.

با گسترش شبکه و اینترنت، نیاز به زبانی احساس شد که برنامه های آن بتوانند برروی هر ماشین و هر سیستم عامل دلخواهی اجرا گردد. شرکت سان مایکروسیستمز در سال 1995 میلادی زبان Java را برمبنای C و++ Cایجاد کرد که هم اکنون از آن در سطح وسیعی استفاده می شود و برنامه های نوشته شده به آن برروی هر کامپیوتری که از Java پشتیبانی کند(تقریبا تمام سیستمهای شناخته شده) قابل اجرا می باشد. شرکت مایکروسافت در رقابت با شرکت سان، در سال 2002 زبان جدیدی بنام# C (سی شارپ) را ارائه داد که رقیبی برای Java بشمار می رود.

برنامه نویسی ساخت یافته

در دهه 1960 میلادی توسعه نرم افزار دچار مشکلات عدیده ای شد. در آن زمان سبک خاصی برای برنامه نویسی وجود نداشت و برنامه ها بدون هیچگونه ساختار خاصی نوشته می شدند. وجود دستور پرش (goto) نیز مشکلات بسیاری را برای فهم و درک برنامه توسط افراد دیگر ایجاد می کرد، چرا که جریان اجرای برنامه مرتبا دچار تغییر جهت شده و دنبال کردن آن دشوار می گردید. لذا نوشتن برنامه ها عملی بسیار زمان بر و پرهزینه شده بود و معمولا اشکال

زدایی, اعمال تغییرات و گسترش برنامه ها بسیار مشکل بود. فعالیتهای پژوهشی در این دهه باعث بوجود آمدن سبک جدیدی از برنامه نویسی بنام روش ساختیافته گردید؛ روش منظمی که باعث ایجاد برنامه هایی کاملا واضح و خوانا گردید که اشکال زدایی و خطایابی آنها نیز بسیار ساده تر بود.

اصلی ترین نکته در این روش عدم استفاده از دستور پرش (goto) است. تحقیقات بوهم و ژاکوپینی نشان داد که می توان هر برنامه ای را بدون دستور پرش و فقط با استفاده از 3 ساختار کنترلی ترتیب، انتخاب و تکرار نوشت.

ساختار ترتیب، همان اجرای دستورات بصورت متوالی (یکی پس از دیگری) است که کلیه زبانهای برنامه نویسی در حالت عادی بهمان صورت عمل می کنند.

ساختار انتخاب به برنامه نویس اجازه می دهد که براساس درستی یا نادرستی یک شرط، تصمیم بگیرد کدام مجموعه از دستورات اجرا شود.

ساختار تکرار نیز به برنامه نویسان اجازه می دهد مجموعه خاصی از دستورات را تا زمانیکه شرط خاصی برقرار باشد، تکرار نماید.

(برای شرح بیشتر موارد فوق به فصل 3 مراجعه نمایید).

هر برنامه ساخت یافته از تعدادی بلوک تشکیل می شود که این بلوکها به ترتیب اجرا می شوند تا برنامه خاتمه یابد(ساختار ترتیب). هر بلوک می تواند یک دستورساده مانند خواندن، نوشتن یا تخصیص مقدار به یک متغیر باشد و یا اینکه شامل دستوراتی باشد که یکی از 3 ساختار فوق را پیاده سازی کنند. نکته مهم اینجاست که درمورد دستورات داخل هر بلوک نیز همین قوانین برقرار است و این دستورات می توانند از تعدادی بلوک به شرح فوق ایجاد شوند و تشکیل ساختارهایی مانند حلقه های تودرتو را دهند.

نکته مهم اینجاست که طبق قوانین فوق یک حلقه تکرار یا بطور کامل داخل حلقه تکرار دیگر است و یا بطور کامل خارج آن قرار می گیرد و هیچگاه حلقه های روی هم افتاده نخواهیم داشت.

از جمله اولین تلاشها در زمینه ساخت زبانهای برنامه نویسی ساخت یافته، زبان پاسکال بود که توسط پروفسور نیکلاس ویرث در سال 1971 برای آموزش برنامه نویسی ساخت یافته در محیطهای آموزشی ساخته شد و بسرعت در دانشگاهها رواج یافت. اما بدلیل نداشتن بسیاری از ویژگیهای مورد نیاز مراکز صنعتی و تجاری در بیرون دانشگاهها موفقیتی نیافت.

کمی بعد زبان C ارائه گردید که علاوه بر دارا بودن ویژگیهای برنامه نویسی ساخت یافته بدلیل سرعت و کارایی بالا مقبولیتی همه گیر یافت و هم اکنون سالهاست که بعنوان بزرگترین زبان برنامه نویسی دنیا شناخته شده است.

مراحل اجرای یک برنامه C

برای اجرای یک برنامه C ابتدا باید آن را نوشت. برای اینکار می توان از هر ویرایشگر متنی موجود استفاده کرد و سپس فایل حاصل را با پسوند .C ذخیره نمود (فایلهای C++ با پسوند .CPP ذخیره می گردند). به این فایل، کد مبدا (source code) گفته می شود. مرحله بعدی تبدیل کد مبدا به زبان ماشین است که به آن کد مقصد (object code) گفته می شود. همانطورکه قبلا نیز گفته شد برای اینکار از یک برنامه مترجم بنام کامپایلر استفاده می شود. کامپایلرهای متعددی برای زبان C توسط شرکتهای مختلف و برای سیستم عاملهای مختلف نوشته شده است که می توانید برحسب نیاز از هریک از آنها استفاده نمایید. اما هنوز برنامه برای اجرا آماده نیست. معمولا برنامه نویسان از در برنامه های خود از یک سری از کدهای از پیش آماده شده برای انجام عملیات متداول (مانند محاسبه جذر و یا سینوس) استفاده می کنند که برنامه آنها قبلا نوشته و ترجمه شده است. این برنامه ها یا در قالب کتابخانه های استاندارد توسط شرکتهای ارائه کننده نرم افزار عرضه شده است و یا توسط دیگر همکاران برنامه نویس اصلی نوشته و در اختیار وی قرار داده شده است. در این مرحله باید کد مقصد برنامه اصلی با کدهای مربوط به این برنامه های کمکی پیوند زده شود. برای اینکار نیاز به یک پیوند زننده (Linker) داریم و نتیجه این عمل یک فایل قابل

اجرا خواهد بود (در ویندوز این فایل پسوند .EXE خواهد داشت). مرحله بعدی اجرای برنامه و دادن ورودیهای لازم به آن و اخذ خروجیها می باشد. در شکل زیر این مراحل نشان داده شده اند.

مسلما طی مراحل بالا برای اجرای هر برنامه زمانبر می باشد، بهمین دلیل اکثر تولید کنندگان کامپایلرها، محیطهایی را برای برنامه نویسی ارائه کرده اند که کلیه مراحل بالا را بطور اتوماتیک انجام می دهند.

به این محیطها (IDE Integrated Development Environment) یا محیط مجتمع توسعه نرم افزار گفته می شود. این محیطها دارای یک ویرایشگر متن می باشند که معمولا دارای خواص جالبی همچون استفاده از رنگهای مختلف برای نشان دادن اجزای مختلف برنامه مانند کلمات کلیدی، و یا قابلیت تکمیل اتوماتیک قسمتهای مختلف برنامه می باشد. پس از نوشتن برنامه و با انتخاب گزینه ای مانند Run کلیه عملیات فوق بطور اتوماتیک انجام شده و برنامه اجرا می گردد. علاوه براین، این محیطها معمولا دارای امکانات اشکالزدایی برنامه (Debug) نیز می باشند که شامل مواردی همچون اجرای خط به خط برنامه و یا دیدن محتویات متغیرها در زمان اجرا است. چند محیط معروف برنامه نویسی عبارتند از :

- Borland C++ 3.1 برای محیط DOS
- Borland C ++از نسخه 4 به بالا برای Windows
- Microsoft Visual C ++ برای محیط Windows
- Borland C++ Builder برای محیط Windows

برای شروع ما از محیط Borland C++ 3.1 تحت Dos که نحوه کار ساده تری نسبت به سایرین دارد استفاده می کنیم.

پس از نصب این نرم افزار، برنامه BC.exe را اجرا کنید تا وارد محیط borland c شوید همانطور که می بینید، این محیط از 3 قسمت اصلی تشکیل شده است :

بخش ویرایش برنامه : بخش آبی رنگ وسط می باشد که در حقیقت یک ویرایشگر است که برنامه در آن تایپ می شود. همانطور که می بینید در این ویرایشگر از رنگهای مختلف برای نشان دادن قسمتهای مختلف برنامه استفاده می شود. مثلا برای کلمات کلیدی از رنگ سفید استفاده شده است.
بخش منوهای کاری : این بخش که در قسمت بالا واقع شده است، حاوی تعدادی منو (گزینه) برای انجام وظایف مختلف است.

خطاهای برنامه نویسی

بنظر می رسد خطاها جزء جداناپذیر برنامه ها هستند. بندرت می توان برنامه ای نوشت که در همان بار اول بدرستی و بدون هیچگونه خطایی اجرا شود. اما خطاها از لحاظ تاثیری که بر اجرای برنامه ها می گذارند، متفاوتند. گروهی ممکن است باعث شوند که از همان ابتدا برنامه اصلا کامپایل نشود و گروه دیگر ممکن است پس از گذشت مدتها و در اثر دادن یک ورودی خاص به برنامه، باعث یک خروجی نامناسب و یا یک رفتار دور از انتظار (مانند قفل شدن برنامه) شوند. بطور کلی خطاها به دو دسته تقسیم می شوند:

خطاهای نحوی (خطاهای زمان کامپایل): این خطاها در اثر رعایت نکردن قواعد دستورات زبان C و یا تایپ اشتباه یک دستور بوجود می آیند و در همان ابتدا توسط کامپایلر به برنامه نویس اعلام می گردد. برنامه نویس باید این خطا را رفع کرده و سپس برنامه را مجددا کامپایل نماید. لذا معمولا این قبیل خطاها خطر کمتری را در بردارند.

خطاهای منطقی (خطاهای زمان اجرا): این دسته خطاها در اثر اشتباه برنامه نویس در طراحی الگوریتم درست برای برنامه و یا گاهی در اثر درنظر نگرفتن بعضی شرایط خاص در برنامه ایجاد می شوند. متاسفانه این دسته خطاها در زمان کامپایل اعلام نمی شوند و در زمان اجرای برنامه خود را نشان می دهند. بنابراین، این خود برنامه نویس است که پس از نوشتن برنامه باید آن را تست کرده و خطاهای منطقی آن را پیدا کرده و رفع نماید. متاسفانه ممکن است یک برنامه نویس خطای منطقی برنامه خود را تشخیص ندهد و این خطا پس از مدتها و تحت یک شرایط خاص توسط کاربر برنامه کشف شود. بهمین دلیل این دسته از خطاها خطرناکتر هستند. خود این خطاها به دو دسته تقسیم می گردند:

- **a. خطاهای مهلک: در این دسته خطاها کامپیوتر بلافاصله اجرای برنامه را متوقف کرده و خطا را به کاربر گزارش می کند. مثال معروف این خطاها،**
- **b. خطای تقسیم بر صفر می باشد.**
- **c. خطاهای غیرمهلک: در این دسته خطا**
- **d. اجرای برنامه ادامه می یابد ولی برنامه نتایج اشتباه تولید می نماید. بعنوان مثال ممکن است دراثر وجود یک خطای منطقی در یک برنامه حقوق و دستمزد**
- **e. حقوق کارمندان اشتباه محاسبه شود و تا مدتها نیز کسی متوجه این خطا نشود!**

با توجه به آنچه گفته شد، در می یابیم که رفع اشکال برنامه ها بخصوص خطاهای منطقی از مهمترین و مشکلترین وظایف یک برنامه نویس بوده و گاهی حتی سخت تر از خود برنامه نویسی است! بهمین دلیل است که بسیاری از شرکتها(همانند مایکروسافت) ابتدا نسخه اولیه نرم افزار خود را در اختیار کاربران قرار می دهند تا اشکالات آن گزارش شده و رفع گردد. بسیار مهم است که در ابتدا سعی کنید برنامه ای بنویسید که حداقل خطاها را داشته باشد، در گام دوم با آزمایش دقیق برنامه خود هرگونه خطای احتمالی را پیدا کنید و در گام سوم بتوانید دلیل بروز خطا را پیدا کرده و آنرا رفع نمایید. هر سه عمل فوق کار سختی بوده و نیاز به تجربه و مهارت دارد.

آخرین نکته اینکه در اصطلاح برنامه نویسی به هر گونه خطا، bug و به رفع خطا debug گفته می شود.

درمورد برنامه فوق به نکات زیر توجه کنید :

خط اول یک توضیح درمورد برنامه است. در زبان C برای توضیحات یک خطی از علامت // استفاده می گردد. اما چنانچه توضیحات بیش از یک خط بود، آن را با علامت /* شروع کرده و با */ پایان دهید. کامپایلر از این توضیحات صرفنظر خواهد کرد. این توضیحات باعث می شوند که برنامه شما خواناتر شده و دیگران بهتر آن را درک کنند.

هر دستوری که با علامت # شروع شود، یک دستور C نیست، بلکه جزو دستورات پیش پردازنده محسوب می گردد. دستورات پیش پردازنده، دستوراتی هستند که توسط کامپایلر قبل از شروع به کامپایل انجام می شوند. بعنوان مثال دستور #include باعث می شود که تعاریف اولیه مربوط به توابعی (زیربرنامه هایی) که قصد استفاده از آنها را داریم به برنامه اضافه شود. در مثال بالا برای استفاده از توابع printf و scanf که در کتابخانه استاندارد C تعریف شده اند، فایل سرآمد stdio.h را که این توابع در آن تعریف شده اند را استفاده کرده ایم.

هر برنامه C باید دارای تابعی به نام main باشد که اجرای برنامه از آن شروع می شود و در حقیقت همان برنامه اصلی است. البته می توان هر تعداد دیگری تابع (زیربرنامه) نیز تعریف کرد، اما وجود تابع main الزامی است. دقت کنید که گرچه این تابع پارامتر ورودی ندارد، اما از پرانتز باز و بسته تنها استفاده شده است.

در زبان C هر بلوک برنامه با علامت { آغاز شده و با } پایان می یابد. این دو معادل دستورات begin و end در زبانهای دیگر ازجمله پاسکال می باشند که برای سادگی زبان انتخاب شده اند.

دو خط بعدی به تعریف متغیرهای radius و area می پردازد. در زبان C قبل از استفاده از هر متغیری باید آن را اعلان نمایید. اعلان متغیر شامل نام و نوع متغیر است. در مثال فوق، متغیر radius از نوع عدد صحیح(integer) و متغیر area از نوع عدد اعشاری (float) تعریف شده اند.

توابع printf و scanf جزو کتابخانه استاندارد C محسوب می گردند و به ترتیب برای چاپ اطلاعات در خروجی استاندارد (نمایشگر) و دریافت اطلاعات از ورودی استاندارد (صفحه کلید) استفاده می شوند. برای چاپ رشته مورد نظر باید آنها را در داخل علامت " قرار داد. درمورد این توابع بعداد توضیح خواهیم داد.

دقت کنید که در پایان هر دستورالعمل از علامت ; استفاده شده است. در مجموع C یک زبان قالب آزاد است و شما می توانید دستورات را به هر نحوی که دوست دارید قرار دهید (مثلا چند دستور در یک خط از برنامه). تنها چیزی که نشاندهنده پایان یک دستور است،- علامت ; است و نه انتهای خط.

از آنجا که C یک زبان قالب آزاد است،- می توان با استفاده از مکان نوشتن دستورات شکل بهتری به برنامه داد. بعنوان مثال دقت کنید که پس از شروع تابع main،- دستورات حدود 3 کاراکتر جلوتر نوشته شده اند. به این نحوه نوشتن دستورات دندانه گذاری می گویند. بطور کلی هربار که بلوک جدیدی آغاز می شود،- باید آن را کمی جلوتر برد. این مسئله باعث جدا شدن بلوکها از یکدیگر و خوانایی بهتر برنامه می شود.

در پایان برنامه و در داخل مستطیل خاکستری،- یک نمونه از اجرای برنامه که شامل یک ورودی و خروجی نمونه است،- آورده شده است.

فصل چهارم: Visual Basic 6

مقدمه ای بر زبان برنامه نویسی Visual Basic 6

ویژوال بیسیک توسعه یافته زبان برنامه‌نویسی بیسیک می‌باشد. بیسیک توسط پروفسور جان کمنسی و توماس کرتز از کالج دارتموث برای نوشتن برنامه‌های ساده ایجاد شد. طراحی آن از اواسط دهه ۱۹۶۰ آغاز گردید.

ویژوال بیسیک تا نسخه ۳ به صورت ۱۶ بیتی بود. از نسخه ۵ به بعد فقط ویرایش ۳۲ بیتی آن ارائه شد. (نسخه ۴ هم به صورت ۱۶ بیتی و هم به صورت ۳۲ بیتی عرضه شده بود.)

ویژوال بیسیک از نسخه ۶ به بعد بر پایه چارچوب دات‌نت (.NET) ارائه شد.

اگر چه با ظهور ویژوال بیسیک دات نت اکثر برنامه‌نویسان ویژوال بیسیک ۶ به آن گرویدند، ولی نسخه ۶ همچنان طرفداران ویژه خود را دارد.

ویژوال بیسیک برای توسعه سریع نرم‌افزار (RAD یا Rapid Application Development) بر پایه رابط گرافیگی کاربر (GUI یا Graphical User Interface) توسعه داده شد. دسترسی آسان و سریع به پایگاه داده‌ها با استفاده از RDO،DAO یا ADO و ایجاد کنترل‌های اکتیو ایکس از جمله مواردی هستند که این زبان را برای RAD مناسب کرده‌اند.

برنامه‌نویسی در ویژوال بیسیک به صورت رخدادگرا و شئ گرا می‌باشد.

در برنامه‌نویسی تجاری، ویژوال بیسیک جز محبوب‌ترین‌ها است. بنابه آماری که در سال ۲۰۰۳ منتشر شد، ٪۵۳ از برنامه‌های تجاری با استفاده از این زبان تولید شده‌اند.visual Basic (که زین پس آن را VB خواهیم خواند) از زبانهای برنامه نویسی تحت Windows می باشد که برای کدنویسی از دستورات زبان Basic سود می برد.

VB (مانند تمام زبانهای برنامه نویسی تحت ویندوز) با استفاده از تمام امکانات زیبای ویندوز (که باعث فراگیرشدن این سیستم عامل زیبا و توانمند در میان کاربران شده است)، طراحی محیطی زیبا و قدرتمند را برای پروژه مورد نظر، بسیار ساده می نماید.

در حال حاضر، به جرأت می توان گفت که یکی از انتخاب های اصلی برنامه نویسان حرفه ای در سطح جهان برای تهیه پروژه های با قابلیت های ویژه، VB می باشد، خصوصاً از VB6 که مایکروسافت عملاً با افزودن توانایی های متنوع بسیار به VB، حتی حاضر شد از دیگر زبانهای معروف تحت ویندوز خود مانند Visual C++و Visual Foxpro بگذرد و سعی در هدایت تمام برنامه نویسان به سوی VB داشته باشد. در حال حاضر که به تهیه این قسمت همت گماردم، نسخه آزمایشی VB.net در بازار وجود دارد ولی عموماً برنامه نویسان ازVB6 استفاده می کنند و لذا ما نیز این نسخه ازVB را برای آموزش انتخاب می کنیم، هر چند تمام خوانندگان عزیز می دانند که عموماً با فراگیری یک نسخه از یک برنامه، فراگیری نسخه های بعدی آن کار چندان دشواری نخواهد بود.

خوانندگان عزیز توجه داشته باشند که برای فراگیری VB، آشنایی با ویندوز و Basic لازم است. همچنین بهتر است در هنگام مطالعه، VB باز باشد و مطالب را در همان لحظه کار کنند.

هنگامی که VB اجرا می شود، توسط پنجره ای نوع پروژه ای که می خواهید طراحی کنید از شما خواسته می شود.

همان طور که ملاحظه می فرمایید، انواع مختلفی از پروژه ها در این پنجره وجود دارد که در ادامه به برخی از آنها خواهیم پرداخت، ولی فعلاً نوع استاندارد پروژه های VB یعنی نوع Standard EXE را انتخاب و آن را باز(Open) می کنیم. این نوع پروژه (که غالب پروژه ها را در بر می گیرد) برای تهیه برنامه های کاربردی(Applications) مورد استفاده قرار می گیرد.

پس از باز کردن پروژه Standard Exe ، پنجره اصلی VB برای این نوع پروژه باز می شود: این پنجره، علاوه بر آنچه عموماً در پنجره های ویندوز می بینیم (مانند Title Bar وMenu Bar)، شامل چند بخش بسیار مهم می باشد:

- 1- یک فرم(Form) خالی با عنوان (Caption)برابر Form1 وجود دارد. این همان فرمی است که بلافاصله پس از اجرا (Run) شدن برنامه، روی صفحه نمایش، قرار می گیرد (این پیش فرض قابل تغییر است).
- 2- نوار ابزار Standard Buttons که دکمه هایی با کاربرد معمولاً بیشتر را شامل می شود. تعدادی از این دکمه ها را در ویندوز می شناسید (مانند Copy, Paste, Undo, Redo, Open وSave) و برخی دیگر را در ادامه خواهیم دید.
- 3- پنجره ای سمت چپ تصویر دیده می شود. این پنجره شامل برخی از کاربردی ترین کنترلهای قابل دیدن (VCL) می باشد. در ادامه با VCLها آشنا خواهیم شد و از آنها بسیار بهره خواهیم برد. به این پنجره Tool Bar (جعبه ابزار) گفته می شود.
- 4- در سمت راست تصویر سه پنجره دیگر دیده می شود. در بالا، پنجره پروژه (Project)، سپس پنجره مشخصات(Properties) و در پایین، پنجرهForm Layout قرار دارد. پنجرهProject شامل نام تمام اجزای پروژه مانند فرمها، ماژولها Moduls،Activex ها و ... می باشد. مثلاً اگر در پروژه ای چند فرم وجود داشته باشد و بخواهیم به فرم دیگری برویم، کافی است نام آن را در این پنجره دابل کلیک نماییم. پنجره Properties، برخی از مشخصه(Property) های مربوط به VCL ای که انتخاب شده (Select) باشد را نمایش می دهد که می توان آنها را در هنگام طراحی(Design Time) تغییر داد (سایر مشخصه ها باید در هنگام اجرا(Run Time) تنظیم شوند.) در پنجرهForm Layout نیز می توان

مکان قرار گیری Form هنگام اجرای برنامه(Run Time) بر روی صفحه نمایش را تعیین کرد (همچنین این کار را توسط کدنویسی نیز می توان انجام داد و عموماً همین روش هم توصیه می شود و لذا عموماً حتی می توان این پنجره را به کل بست).

در این بخش با شمای کلی VB آشنا شدیم، در ادامه درباره VCL ها و نحوه کدنویسی برای آنها مطالب مفیدی خواهیم آموخت.

مفاهیم بنیادی در VB، شئ (Object) های بسیاری وجود دارد مانند فرمها، دکمه ها، برچسب ها، تصاویر و همانطور که می دانیم هر شئ(Object) دارای یک سری مشخصات(Properties) می باشد. به عنوان مثال اگر یک اتومبیل را به عنوان یک شئ در نظر بگیریم، این اتومبیل دارای مشخصاتی چون رنگ خاص، وزن خاص، طول و عرض و ارتفاع خاص، میزان خاص مصرف بنزین و ... می باشد که در تمایز دو اتومبیل از هم، همین مشخصات هستند که به ما کمک می کنند.

درVB نیز هر Object دارای یک تعداد مشخصه (Property) می باشد. به عنوان مثال یک دکمه(Button) دارای مشخصاتی چون عرض(Width) و ارتفاع(Height) خاص و یا یک عنوان(Caption) خاص و ... می باشد.
برخی شئ(object) ها در VB، فقط در کدنویسی قابل دسترسی هستند(مانند شئADODB که در آینده با آن آشنا خواهید شد) اما برخی دیگر علاوه بر زمان کدنویسی، در زمان طراحی (Design) نیز می توان آنها را بر روی فرمها و در جای دلخواه قرار داد و آنها را تنظیم (Set) نمود. به اشیاء نوع اخیر، کنترل (Control) گفته می شود.

کنترل ها خود دو گونه اند، برخی علاوه بر زمان طراحی (Design) در زمان اجرا (Run Time) نیز دیده می شوند، به این نوع کنترل در اصطلاح Visual Control VCL گفته می شود که بیشترین انواع کنترلها را در بر می گیرند (مانند دکمه ها، جدولها، برچسبها و بسیاری دیگر که در ادامه خواهند آمد) اما برخی دیگر از کنترل ها فقط در هنگام طراحی (Design) دیده می شوند و در هنگام اجرا تنها عمل خاصی انجام می دهند و خود دیده نمی شوند (مانند کنترلTimer)، به این نوع کنترلها، Non-Visual Control گفته می شود. باید توجه داشته باشیدکه کنترلهایNon-Visual، ذاتاً در Run Time دیده نمی شوند ولی ممکن است بنابر نیازی و در زمانی خاص از اجرا، خودمان برای یک یا چند VCL نیز مقدار مشخصه Visual آنها را برابرFalse قرار دهیم که مسلماً در این حالت با وجود اینکه در آن زمان، این کنترلها دیده نمی شوند ولی VCL بودن آنها تغییری نکرده است.

کنترل ها (اعم از VCL ها و غیر آن) معمولاً دارای تعدادی Event می باشند. Event ، رویداد یا رخدادی است که توسط کاربر و معمولاً با استفاده از ماوس یا صفحه کلید برای یک کنترل خاص روی می دهد.مثلاً یک کنترل ممکن است دارای رویداد Event Click باشدکه این نوع رویداد زمانی که کاربر در هنگام اجرای برنامه (Run Time) بر روی آن کنترل خاص کلیک نماید رخ می دهد.برخی Event های معمول دیگر برای کنترلها عبارتند از: DoubleClick (زمانی که بر روی آن کنترل دابل کلیک شود) ، MouseMove (زمانی که نشانگر ماوس بر روی آن کنترل قرار دارد) ، KeyPress (زمانی که کلیدی از صفحه کلید زده شد) ، KeyDown (زمانی که کلیدی از صفحه کلید پایین بود) ، KeyUp (زمانی که کلید زده شده برداشته شد) ، MouseDown، MouseUp و ... که در ادامه با آنها بیشتر آشنا خواهیم شد.
برای هر Event (رویداد) می توان یک Event Procedure داشت. Event Procedure پاسخی است که یک کنترل زمانی که یک Event رخ می دهد، از خود نشان می دهد. در واقع Event قطعه برنامه ای است که زمانی که بر روی یک کنترل، یک Event رخ می دهد، به طور اتوماتیک اجرا می شود.

کنترلها معمولاً علاوه بر یک سری مشخصات (Properties) و یک سری رویداد (Event) ، دارای تعدادی نیز متد (Method) می باشد. متدها عملیاتهای تعریف شده ای هستند که توسط آنها یک عمل خاص بر روی کنترلها انجام می شود.

توجه داریم که تفاوت متدها و Event Procedure ها در این است که متدها توسط VB ، شناخته شده اند و عملشان همیشه ثابت است، اما Event Procedure ها توسط برنامه نویس و به دلخواه او تهیه می شود، بنابراین Method های مشابه بر روی کنترلهای متفاوت، پاسخ مشابهی دارد ولی ممکن است

Event های مشابه بر روی کنترلهای متفاوت با توجه به Event Procedure های مخصوص هر یک، متفاوت باشد (مثلاً رویداد Click برای یک کنترل، کاری انجام دهد و برای کنترلی دیگر، کاری دیگر).

نکته دیگری که باید به آن توجه داشت اینست که Procedure ها به طور مستقیم اجرا نمی شوند بلکه فقط زمانی که نامشان فراخوانی شود اجرا می شوند. بنابراین مثلاً در مورد Event Procedure ها باید بدانیم که با اینکه کد مربوط به آنها نوشته شده است ولی تا وقتی که آن Event خاص (که باعث فرا خوانی Event Procedure مربوط می شود) روی ندهد، این کدها اجرا نخواهند شد.

فصل پنجم: زبان Java

جاوا (زبان برنامه‌نویسی)

جاوا (به انگلیسی: Java) یک زبان برنامه‌نویسی شئ‌گراست که برای اولین بار توسط جیمز گوسلینگ در شرکت سان مایکروسیستمز ایجاد شد و در سال ۱۹۹۵ به عنوان بخشی از سکوی جاوا منتشر شد. زبان جاوا شبیه به C++ است اما مدل شیءگرایی آسان‌تری دارد و از قابلیت‌های سطح پایین کمتری پشتیبانی می‌کند. یکی از قابلیت‌های اصلی جاوا این است که مدیریت حافظه را بطور خودکار انجام می‌دهد. ضریب اطمینان عملکرد برنامه‌های نوشته‌شده به این زبان بالا است و وابسته به سیستم‌عامل خاصی نیست، به عبارت دیگر می‌توان آن را روی هر رایانه با هر نوع سیستم‌عاملی اجرا کرد. برنامه‌های جاوا به صورت کدهای بیتی همگردانی (کامپایل) می‌شوند. که مانند کد ماشین هستند و به ویژه وابسته به سیستم‌عامل خاصی نیستند.

تاریخچه

در مقایسه با زبان‌های دیگر، همچون C++ یا بیسیک یا فورترن، جاوا زبان نسبتاً جدیدتری است. شرکت (sun Microsystems)سان مایکروسیستمز در سال ۱۹۹۱ یک پروژه تحقیقاتی به نام گرین (Green) را آغاز کرد. هدف این پروژه ایجاد زبانی جدید شبیه به C++ بود که نویسنده اصلی آن، جیمز گاسلینگ، آن را بلوط (Oak) نامید. اما بعدها به دلیل برخی مشکلات حقوقی از میان لیستی از کلمات تصادفی[1] نام آن به جاوا تغییر کرد.

پروژه گرین به دلیل مشکلات بازاریابی در شرف لغو شدن بود تا اینکه گسترش وب در سال ۱۹۹۳ باعث نمایش توانایی‌های وافر جاوا در این عرصه گشت. اینگونه بود که شرکت سان مایکروسیستمز در مه ۱۹۹۵ جاوا را رسماً به بازار عرضه کرد.

جاوا یک زبان برنامه‌نویسی است که در ابتدا توسط شرکت ایجاد شده‌است و در سال 1995 به عنوان مولفه اصلی java platform منتشر شد.این زبان قسمت‌های بسیاری از گرامر خود را از C و C++ گرفته اما دارای مدل شیگرایی ساده‌ای است و امکانات سطح پایین کمی دارد. کاربرد جاوا در کامپایل به صورت بایت کد است که قابلیت اجرا روی تمامی ماشین‌های شبیه‌سازی جاوا را داشته باشد صرف نظر از معماری و خصوصیات آن کامپیوتر. اجرای اصلی کامپایلرهای جاوا، ماشین‌های پیاده‌سازی و کتابخانه‌های آن توسط این شرکت از سال 1995 منتشر شد. در 2007 may این شرکت، نرم‌افزار رایگان این زبان را فراهم کرد. دیگران هم کاربردهای دیگری از این زبان را منتشر کردند مثل کامپایلر GNU برای جاوا.

مرورگرهای اصلی وب، به هم پیوستند تا به طور مطمئن java applet را بدون صفحات وب اجرا کنند و به این صورت جاوا خیلی زود معروف و محبوب شد. با پیدایش java2، نسخه جدید توانست ترکیب‌های جدیدی را برای نوع‌های مختلف پلت فرم‌ها ایجاد کند. به عنوان مثال J2EE باهدف کاربرد برای تشکیلات اقتصادی، و نسخه J2ME برای موبایل منتشر شد. در سال 2006 با هدف بازاریابی، این شرکت نسخه جدید J2 را با نام‌های JavaEE JavaME و JavaSE منتشر کرد. در سال 1997 شرکت سان میکروسیستمز،ISO/IEC JTC1 standards body و Ecma International را به فرمول جاوا تغییر داد. شرکت sun بسیاری از کاربردهای جاوایش را بدون هیچ هزینه‌ای فراهم آورد. شرکت sun با فروش مجوز برای بعضی از کاربردهای خاصش مثل Java Enterprise System درآمدی را بدست آورد. اولین تمایزی که بین SDK و JRE داد شامل فقدان کامپایلر

برای JRE و سرفایل‌ها بود. در 13 نوامبر 2006 شرکت sun نرم‌افزار جاوا را به صورت رایگان و با مجوز عمومی برای همه منتشر کرد. جاوا یک پلتفرم نرم‌افزاری است.

اهداف اولیه

این زبان باید ساده، شی‌گرا و مشهور باشد.

مطمئن و بدون خطا باشد.

وابسته به معماری کامپیوتر نبوده و قابل انتقال باشد.

باید با کارایی بالا اجرا شود.

باید به صورت پویا و نخ‌کشی‌شده باشد.

برنامه‌های جاوا و اپلت‌ها

جاوا برای نوشتن انواع برنامه‌های کاربردی مناسب است. با جاوا می‌توان انواع برنامه‌های زیر را نوشت:

برنامه‌های تحت وب

برنامه‌نویسی سیستم‌های کوچک مانند موبایل، پاکت پی‌سی و ...

برنامه‌های کاربردی بزرگ (Enterprise)

برنامه‌های رومیزی (Desktop)

و غیره.

قابلیت خاصی در جاوا وجود دارد بنام اپلت. اپلت‌ها امکانات فراوانی برای نوشتن برنامه‌های تحت وب در اختیار برنامه‌نویسان قرار می‌دهند که دیگر زبان‌های برنامه‌نویسی فاقد آن هستند. البته وجود ماشین مجازی جاوا برای اجرای اپلت لازم است. اپلت‌ها نظیر فناوری Activex شرکت مایکروسافت هستند که برنامه‌نویسان را قادر می‌سازد تا امکاناتی را به مرورگر کاربر بیافزایند.البته تفاوت این دو در امنیت می‌باشد به گونه‌ای که اپلت‌ها بدلیل اینکه در محیطی به نام جعبه شنی اجرا می‌شوند امن هستند ولی Activex ها فاقد چنین امنیتی هستند.

سیستم‌عامل: هر چقدر زبانهای .net قوی باشند تنها بر روی پلت فرم ویندوز اجرا می‌شوند و برخی ویندوز را سیستم‌عامل غیر قابل اعتمادی در برنامه نویسی Enterprise می‌دانند. ولی جاوا از این نظر انتخابی خوب است.

قابلیت حمل: جاوا بر روی پلتفرم‌های گوناگونی قابل اجرا است، از ATM و ماشین رختشویی گرفته تا سرورهای سولاریس با قابلیت پشتیبانی از cpu 1024 برای پردازش.

جاوا بیشتر از یک زبان است: جاوا فقط یک زبان نیست و انجمن‌هایی متشکل از بزرگان صنایع و برنامه‌نویسان زیادی مشغول به توسعه و ایجاد استانداردهای جدید و به روز هستند.

- **خط مشی جاوا**

یکی از ویژگی‌های جاوا قابل حمل بودن آن است. یعنی برنامه نوشته شده به زبان جاوا باید به طور مشابهی در کامپیوترهای مختلف با سخت‌افزارهای متفاوت اجرا شود. و باید این توانایی را داشته باشد که برنامه یک بار نوشته شود، یک بار کامپایل شود و در همه کامپیوترها اجرا گردد. به این صورت که کد کامپایل شده جاوا را ذخیره می‌کند، اما نه به‌صورت کد ماشین بلکه به‌صورت بایت کد جاوا. دستورالعمل‌ها شبیه کد ماشین هستند، اما با ماشین‌های مجازی که به طور خاص برای سخت‌افزارهای مختلف نوشته شده‌اند، اجرا می‌شوند. در نهایت کاربر از JRE نصب شده روی ماشین خود یا مرورگر وب استفاده می‌کند. کتابخانه‌های استاندارد یک راه عمومی برای دسترسی به ویژگی‌های خاص فراهم می‌کنند. مانند گرافیک، نخ‌کشی و شبکه. در بعضی از نسخه‌های JVM بایت کدها می‌توانند قبل و در زمان اجرای برنامه به کدهای محلی کامپایل شوند. فایده اصلی استفاده از بایت کد، قسمت کردن است. اما ترجمه کلی یعنی برنامه‌های ترجمه شده تقریباً همیشه کندتر از برنامه‌های کامپایل شده محلی اجرا می‌شوند. این شکاف می‌تواند با چند تکنیک خوش‌بینانه که در کاربردهای JVM قبلی معرفی شد، کم شود. یکی از این تکنیک‌ها JIT است که بایت کد جاوا را به کد محلی ترجمه کرده و سپس آن را پنهان می‌کند. در نتیجه برنامه خیلی سریع‌تر نسبت به کدهای ترجمه شده خالص شروع و اجرا می‌شود. بیشتر VMهای پیشرفته، به‌صورت کامپایل مجدد پویا، در آنالیز VM، رفتار برنامه اجرا شده و کامپایل مجدد انتخاب شده و بهینه‌سازی قسمت‌های برنامه، استفاده می‌شوند. کامپایل مجدد پویا می‌تواند کامپایل ایستا را بهینه‌سازی کند. زیرا می‌تواند قسمت hot spot برنامه و گاهی حلقه‌های داخلی که ممکن است زمان اجرای برنامه را افزایش دهند را تشخیص دهد. کامپایل JIT و کامپایل مجدد پویا به برنامه‌های جاوا اجازه می‌دهد که سرعت اجرای کدهای محلی بدون از دست دادن قابلیت انتقال افزایش پیدا کند.

تکنیک بعدی به عنوان کامپایل ایستا شناخته شده است. که کامپایل مستقیم به کدهای محلی است مانند بسیاری از کامپایلرهای قدیمی. کامپایلر ایستای جاوا، بایت کدها را به کدهای شی محلی ترجمه می‌کند.

کارایی جاوا نسبت به نسخه‌های اولیه بیشتر شد. در تعدادی از تست‌ها نشان داده شد که کارایی کامپایلر JIT کاملاً مشابه کامپایلر محلی شد.عملکرد کامپایلرها لزوماً کارایی کدهای کامپایل شده را نشان نمی‌دهند.یکی از پیشرفت‌های بی نظیر در در زمان اجرای ماشین این بود که خطاها ماشین را دچار اشکال نمی‌کردند.علاوه بر این در زمان اجرای ماشینی مانند جاوا وسایلی وجود دارد که به زمان اجرای ماشین متصل شده و هر زمانی که یک استثنا رخ می‌دهد، اطلاعات اشکال زدایی که در حافظه وجود دارد، ثبت می‌کنند.

- **پیاده سازی**

شرکت سان میکروسیستم مجوز رسمی برای پلت فرم استاندارد جاوا را به Microsoft Windows، Linux، و Solaris. داده است. همچنین محیط‌های دیگری برای دیگر پلت فرم‌ها فراهم آورده است. علامت تجاری مجوز شرکت سان میکروسیستم طوری بود که با همه پیاده سازی‌ها سازگار باشد.به علت اختلاف قانونی که با ماکروسافت پیدا کرد، زمانی که شرکت سان ادعا کرد که پیاده سازی ماکروسافت از RMI یا JNI پشتیبانی نکرده و ویژگی‌های خاصی را برای خودش اضافه کرده‌است. شرکت سان در سال 1997 پیگیری قانونی کرد و در سال 2001 در توافقی 20 میلیون دلاری برنده شد. در نتیجه کمی بعدماکروسافت جاوا را به ویندوز فرستاد. در نسخه اخیر ویندوز، مرورگر اینترنت نمی‌تواند از جاوا پلت فرم پشتیبانی کند. شرکت سان و دیگران یک سیستم اجرای جاوای رایگان برای آنها و نسخه‌های دیگر ویندوز فراهم آوردند.

- **اداره خودکار حافظه**

جاوا از حافظه بازیافتی خودکار برای اداره حافظه در چرخه زندگی یک شی استفاده می‌کند. برنامه‌نویس زمانی که اشیا به وجود می‌آیند، این حافظه را تعیین می‌کند. و در زمان اجرا نیز، زمانی که این اشیا در استفاده زیاد طولانی نباشند، برنامه نویس مسئول بازگرداندن این حافظه است. زمانی که مرجعی برای شی‌های

باقیمانده نیست، شی‌های غیر قابل دسترس برای آزاد شدن به صورت خودکار توسط بازیافت حافظه، انتخاب می‌شوند. اگر برنامه‌نویس مقداری از حافظه را برای شی‌هایی که زیاد طولانی نیستند، نگه دارد، چیزهایی شبیه سوراخ حافظه اتفاق می‌افتند.

یکی از عقایدی که پشت سر مدل اداره حافظه خودکار جاوا وجود دارد، این است که برنامه‌نویس هزینه اجرای اداره دستی حافظه را نادیده می‌گیرد. در بعضی از زبان‌ها حافظه لازم برای ایجاد یک شی، به صورت ضمنی و بدون شرط، به پشته تخصیص داده می‌شود. و یا به‌طور صریح اختصاص داده شده و از heap بازگردانده می‌شود. در هر کدام از این راه‌ها، مسئولیت اداره اقامت حافظه با برنامه‌نویس است. اگر برنامه شی را برنگرداند، سوراخ حافظه اتفاق می‌افتد. اگر برنامه تلاش کند به حافظه‌ای را که هم‌اکنون بازگردانده شده، دستیابی پیدا کند یا برگرداند، نتیجه تعریف شده نیست و ممکن است برنامه بی‌ثبات شده و یا تخریب شود. این ممکن است با استفاده از اشاره‌گر مدتی باقی بماند، اما سرباری و پیچیدگی برنامه زیاد می‌شود. بازیافت حافظه اجازه دارد در هر زمانی اتفاق بیفتد. به‌طوری که این زمانی اتفاق می‌افتد که برنامه بی‌کار باشد. اگر حافظه خالی کافی برای تخصیص شی جدید در heap وجود نداشته باشد، ممکن است برنامه برای چند دقیقه متوقف شود. در جایی که زمان پاسخ یا اجرا مهم باشد، اداره حافظه و منابع اشیا استفاده می‌شوند.

جاوا از نوع اشاره‌گر ریاضی C و C++ پشتیبانی نمی‌کند. در جایی که آدرس اشیا و اعداد صحیح می‌توانند به جای هم استفاده شوند. همانند C++ و بعضی زبان‌های شی‌گرای دیگر، متغیرهای نوع‌های اولیه جاوا شی‌گرا نبودند. مقدار نوع‌های اولیه، مستقیماً در فیلدها ذخیره می‌شوند. در فیلدها (برای اشیا) و در پشته (برای توابع)، بیشتر از heap استفاده می‌شود. این یک تصمیم هوشیارانه توسط طراح جاوا برای اجرا است. به همین دلیل جاوا یک زبان شی‌گرای خالص به حساب نمی‌آید.

- **گرامر**

گرامر جاوا خیلی بزرگتر از C++ است. مثل C++ که ترکیب ساختارها و برنامه‌های شی‌گرا می‌باشد، نیست. بلکه زبان جاوا یک زبان شی‌گرای خالص است. همه کدهایی که داخل کلاس نوشته می‌شود و همه چیزهایی که داخل شی است، با استثنائات نوع داده اصلی، که به صورت کلاس نیستند، برای اجرا. جاوا بسیاری از ویژگی‌ها را پشتیبانی می‌کند. از کلاس‌ها برای ساده‌تر کردن زبان و جلوگیری از رخداد خطا.

- **نمونه‌هایی از برنامه‌های جاوا**

در زیر نمونه‌ای از برنامه‌ای که در جاوا نوشته شده‌است آورده شده است. البته برای کامپایل کردن این برنامه بایستی JDK بر روی سیستم مورد نظر نصب شده باشد.

```
public class Test{

        public static void main(String[] args) {

                System.out.println("HelloWorld!");

        }

}
```

برای اجرای برنامه بالا، ابتدا باید یک فایل به نام Test.java ساخته شود و سپس کامپایل شود:

```
$ javac Test.java
```

سپس یک فایل خروجی به نام Test.class دریافت می‌شود. بعد با استفاده از دستور زیر برنامه قابل اجرا است:

```
$ java Test
```

مثال‌ها

برنامه Hello world به این صورت در زبان جاوا می‌تواند نوشته شود:

```
// HelloWorld.java

public class HelloWorld {

        public static void main(String[] args) {

                System.out.println("Hello, World!");

        }

}
```

بر طبق قرارداد فایل هل بعد از کلاس‌های عمومی نام گذاری می‌شوند.سپس باید پسوند java را به این صورت اضافه کرد: Hello world.java .این فایل اول باید با استفاده از کامپایلر جاوا به بایت کد کامپایل شود.در نتیجه فایل Hello world.class ایجاد می‌شود.این فایل قابل اجرا است.فایل جاوا ممکن است فقط یک کلاس عمومی داشته باشد.اما می‌تواند شامل چندین کلاس با دستیابی عمومی کمتر باشد.

کلاسی که به صورت خصوصی تعریف می‌شود ممکن است در فایل .java ذخیره شود.کامپایلر برای هر کلاسی که در فایل اصلی تعریف می‌شود یک کلاس فایل تولید می‌کند.که نام این کلاس فایل همنام کلاس است با پسوند .class

کلمه کلیدی public (عمومی) برای قسمت هایی که می‌توانند از کدهای کلاس‌های دیگر صدا زده بشوند، به کار برده می‌شود. کلمه کلیدی static (ایستا) در جلوی یک تابع، یک تابع ایستا را که فقط وابسته به کلاس است و نه قابل استفاده برای نمونه‌هایی از کلاس، نشان می‌دهد.فقط تابع‌های ایستا می‌توانند توسط اشیا بدون مرجع صدا زده شوند.داده‌های ایستا به متغیر هایی که ایستا نیستند، نمی توانند دسترسی داشته باشند.

کلمه کلیدی void (تهی) نشان می‌دهد که تابع main هیچ مقداری را بر نمی گرداند.اگر برنامه جاوا بخواهد با خطا از برنامه خارج شود، باید system.exit() صدا زده شود. کلمه main یک کلمه کلیدی در زبان جاوا نیست.این نام واقعی تابعی است که جاوا برای فرستادن کنترل به برنامه، صدا می‌زند.برنامه جاوا ممکن است شامل چندین کلاس باشد که هر کدام دارای تابع main هستند.

تابع main باید آرایه‌ای از اشیا رشته‌ای را بپذیرد.تابع main می‌تواند از آرگومان‌های متغیر به شکل (public static void main string...args) استفاده کند که به تابع main اجازه می‌دهد اعدادی دلخواه از اشیا رشته‌ای را فراخوانی کند. پارامتر string[]args آرایه‌ای از اشیا رشته ایست که شامل تمام آرگومان هایی که به کلاس فرستاده می‌شود، است.

چاپ کردن، قسمتی از کتابخانه استاندارد جاوا است.کلاس سیستم یک فیلد استاتیک عمومی به نام out تعریف کرده است.شی out یک نمونه از کلاس printstream است و شامل تعداد زیادی تابع برای چاپ کردن اطلاعات در خروجی استاندارد است.همچنین شامل (println string) برای اضافه کردن یک خط جدید برای رشته فرستاده شده اضافه می‌کند.

- **توزیع‌های جاوا**

منظور از توزیع جاوا پیاده‌سازی‌های مختلفی است که برای کامپایلر جاوا و همچنین مجموعه کتابخانه‌های استاندارد زبان جاوا (JDK) وجود دارد. در حال حاضر چهار توزیع‌کننده عمده جاوا وجود دارند:

سان میکروسیستمز: توزیع کننده اصلی جاوا و مبدع آن می‌باشد. در اکثر موارد هنگامی که گفته می‌شود جاوا منظور توزیع سان می‌باشد.

GNU Classpath: این توزیع از سوی موسسه نرم‌افزارهای آزاد منتشر شده و تقریبا تمامی کتابخانه استاندارد زبان جاوا در آن بدون بهره‌گیری از توزیع شرکت سان از اول پیاده‌سازی شده است. یک کامپایلر به نام GNU Compiler for Java نیز برای کامپایل کردن کدهای جاوا توسط این موسسه ایجاد شده‌است. فلسفه انتخاب نام Classpath برای این پروژه رها کردن تکنولوژی جاوا از وابستگی به علامت تجاری جاوا است بطوریکه هیچ وابستگی یا محدودیتی برای استفاده آن از لحاظ قوانین حقوقی ایجاد نشود و از طرفی به خاطر وجود متغیر محیطی classpath در تمامی محیط‌های احرایی برنامه‌های جاوا، این نام به نوعی تکنولوژی جاوا را برای خواننده القا می‌کند. کامپایلر GNU توانایی ایجاد کد اجرایی (در مقابل بایت کد توزیع سان) را داراست. لازم به ذکر است که در حال حاضر شرکت سان تقریبا تمامی کدهای JDK را تحت مجوز نرم‌افزارهای آزاد به صورت متن باز منتشر کرده است و قول انتشار قسمت بسیار کوچکی از این مجموعه را که به‌دلیل استفاده از کدهای شرکت‌های ثانویه نتوانسته به صورت متن باز منتشر نماید در آینده نزدیک با بازنویسی این کدها داده است.

مایکروسافت # J: این در حقیقت یک توزیع جاوا نیست. بلکه زبانی مشابه می‌باشد که توسط مایکروسافت و در چارچوب .net ارائه شده‌است. انتظار اینکه در سیستم‌عاملی غیر از ویندوز هم اجرا شود را نداشته باشید.

AspectJ: این نیز یک زبان مجزا نیست. بلکه یک برنامه الحاقی می‌باشد که امکان برنامه نویسی Aspect Oriented را به جاوا می‌افزاید. این برنامه توسط بنیاد برنامه‌نویسی جلوه‌گرا و به صورت کدباز ارائه شده‌است.

کلاس‌های خاص

- **Applet (برنامه‌های کاربردی کوچک)**

اپلت جاواها برنامه‌هایی هستند که برای کاربردهایی نظیر نمایش در صفحات وب، ایجاد شده‌اند. واژه import باعث می‌شود کامپایلر جاوا کلاس‌های javaapplet.Applet وjava.awt.Graphics را به کامپایل برنامه اضافه کند. کلاس Hello کلاس Applet را توسعه می‌دهد. کلاس اپلت چارچوبی برای کاربردهای گروهی برای نمایش و کنترل چرخه زندگی اپلت، درست می‌کند. کلاس اپلت یک تابع پنجره‌ای مجرد است که برنامه‌های کوچکی با قابلیت نشان دادن واسط گرافیکی برای کاربر را فراهم می‌کند. کلاس Hello تابع موروثی print Graphics را از سوپر کلاس container باطل می‌کند، برای اینکه کدی که اپلت را نمایش می‌دهد، فراهم کند. تابع paint شی‌های گرافیکی را که شامل زمینه‌های گرافیکی هستند را می فرستد تا برای نمایش اپلت‌ها استفاده شوند. تابع paint برای نمایش "Hello world" تابع drawstring string,int,int را صدا می‌زند.

- **Servlet**

تکنولوژی servlet جاوا گسترس وب را به آسانی فراهم می‌کند.و شامل مکانیزم هایی برای توسعه تابعی سرور وب و برایدسترسی به سیستم‌های تجاری موجود است.servlet قسمتی از javaEE است که به درخواست‌های مشتری پاسخ می‌دهد.

```
// Hello.java

import java.io.*;

import javax.servlet.*;

public class Hello extends GenericServlet {

        public void service(ServletRequest request, ServletResponse response)

        throws ServletException, IOException {

                response.setContentType("text/html");

                final PrintWriter pw = response.getWriter();

                pw.println("Hello, world!");

                pw.close();

         }

}
```

واژه import کامپایلر جاوا را هدایت می‌کند که تمام کلاس‌های عمومی و واسط‌ها را از بسته‌های java.io وjava.servlet را در کامپایل وارد کند.

کلاس Hello کلاس Genericservlet را توسعه می‌دهد. کلاس Genericservlet واسطی برای سرور فراهم می‌کند تا درخواست را به servlet بفرستد و چرخه زندگی servlet را کنترل کند.

- **JSP**

صفحه سرور جاوا قسمتی از سرور javaEE است که پاسخ تولید می‌کند.نوعاً صفحات HTML به درخواست‌های HTTP از مشتری.JSP ها کد جاوا در صفحه HTML را با استفاده از حائل <٪and٪> اضافه می‌کنند.JSP به javaservlet کامپایل می‌شود.

- **Swing**

Swing کتابخانه واسط گرافیکی کاربر است برای پلت فرم javaSE. ابزاری مشابه پنجره، GTK و motif توسط شرکت sun فراهم شده‌اند. این مثال کاربرد swing یک پنجره واحد همراه با Hello world را ایجاد می‌کند.

```
// Hello.java (Java SE 5)

import java.awt.BorderLayout;

import javax.swing.*;

public class Hello extends JFrame {

	public Hello() {

		super("hello");

		setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);

		setLayout(new BorderLayout());

		add(new JLabel("Hello, world!"));

		pack();

	}


	public static void main(String[] args) {

		new Hello().setVisible(true);

	}

}
```

اولین جمله import کامپایلر جاوا را هدایت می‌کندتا کلاس Borderlayout را از بسته java.awt در جاوا به کامپایل اضافه کند. و import دوم همه کلاس‌های عمومی و واسط آن‌ها را از بسته javax.swing اضافه می‌کند. کلاس Hello کلاس Jframe را توسعه می‌دهد.کلاس Jframe یک پنجره با میله عنوان و کنترل بستن است.

زمانی که برنامه آغاز می‌شود، تابع main با JVM صدا زده می‌شود.این یک نمونه جدید از کلاس Hello را ایجاد کرده و با صدا زدن تابع (setvisible boolean) با مقدار true نمایش داده می‌شود.

Generics

قبل از کلاس‌های عمومی برای هر متغیر باید یک نوع خاص تعریف می‌کردیم.به عنوان مثال برای کلاس‌های ظرف این امر مشکل بود زیرا را آسانی برای ایجاد یک container وجود نداشت که نوع‌های خاصی از اشیا را بپذیرد.کلاس‌های عمومی اجازه می‌دهند نوع زمان کامپایل، بدون نیاز به ایجاد تعداد زیادی از container، چک شود.همه آنها کدهای مشابهی دارند.

- کتابخانه‌های کلاس
- کتابخانه‌های جاوا که به صورت بایت کد از کد اصلی کامپایل شده اند، برای پشتیبانی از بعضی از کاربردهای جاوا، توسط JRE منتشر شده است.

مثال هایی از این کتابخانه‌ها عبارتند از:

- کتابخانه‌های مرکزی که شامل:
- کتابخانه‌هایی که برای ساختار داده کاربرد دارند.مثل لیست ها، درخت ها، مجموعه ها، مترجم ها.
- کتابخانه پرداز XML (تجزیه، تغییر شکل، اعتبار)
- کتابخانه‌های موضعی و بین‌المللی
- کتابخانه‌های انتگرال گیری که امکان تایپ کردن توسط سیستم‌های بیرونی را می‌دهند.
- JDBC برای دستیابی به داده ها
- JNDI برای مراجعه و کشف کردن
- RMI & CORBA برای توسعه کاربرد توزیع کردن
- کتابخانه‌های واسط کاربر
- AWT (توابع پنجره‌ای مجرد)که قسمت هایی از GUI را فراهم می‌کنند.
- کتابخانه‌های swing که در AWT ساخته شده اند اما کاربرد هایی از AWT widgetry را فراهم می‌کنند.
- APLها برای ضبط صدا، پردازش و بازنواختی
- کاربردهای وابسته پلت فرم ماشین‌های مجازی جاوا
- Plugins که توانایی اجرا شدن در مرورگرهای وب را به اپلت می‌دهد.
- java web start
- دادن مجوز و مستند سازی
- ویرایش
- شرکت سان میکروسیستم، 4 نوع ویرایش از کاربردهای مختلف جاوا را ارائه داده است:
- Java card for smartcard
- JavaME
- JavaSE

- JavaEE

ایرادات مطرح شده

مهم‌ترین ایرادی که برنامه نویسان سایر زبان‌ها به زبان جاوا می‌گیرند سرعت اجرایی پایین جاوا در مقایسه با زبان‌ها سطح پایین‌تر مانند C++ و اسمبلی است. یک برنامه جاوا به صورت بایت کد می‌باشد و باید در ماشین مجازی جاوا اجرا گردد. به همین دلیل سرعت اجرای پایینی را در مقابل زبان‌هایی همچون C++ دارد. به صورت دیگر یک برنامه C به طور متوسط تا 10 برابر سریعتر از برنامه مشابه جاوا اجرا می‌گردد

جاوا علی رغم شیءگرا بودن در بخشی از قسمت‌ها برخی از اصول شی‌گرایی را نادیده گرفته‌است. از جمله این قسمت‌ها قابلیت بازتابش Reflection می‌باشد. هدف اصلی بازتابش بررسی (مشاهده) و ایجاد تغییر در برنامه در حال اجرا است ولی این مهم با زیر پا گذاشتن بعضی اصول ممکن شده‌است. برای نمونه با استفاده از بازتابش (و در صورت داشتن مجوز لازم ضمن اجرای برنامه) می‌توان به متدهای خصوصی دیگر کلاس‌ها دسترسی داشت.

زبان جاوا در مقابل زبانی مثل C++ ساده تر و یادگیری آن آسانتر است. این آسانتر بودن با حذف بسیاری از موارد که باعث قدرتمندتر بودن زبان C++ بوده‌اند ایجاد شده‌است. مهم‌ترین این موارد اشاره گرها و وراثت چندگانه بوده‌اند که در زبان جاوا یافت نمی‌شوند.

از آنجایی که جاوا زبانی با عدم وابستگی به بستر می‌باشد پس استفاده از توابع سیستم‌عامل در برنامه را به طور مستقیم نمی‌پذیرد. به همین صورت نمی‌توان به طور مستقیم از واسط‌های برنامه نویسی غیر از جاوا در آن استفاده نمود.

پاسخ به ایرادات

سرعت پایین برنامه‌های جاوا در محیطی که اجرا می‌شوند ملاک کارایی نبوده زیرا در محیط وب مسئله‌ای که سرعت را کند می‌سازد، شبکه بوده و ابتدا باید سربار شبکه را از روی برنامه‌ها برداشت. از طرف دیگر در برنامه‌های رومیزی هم در JDK 5.0 و 6.0، بهینه سازی بسیاری بوجود آمده که این مسئله باعث شده که در آخرین تست کارایی که انجام شده یک برنامه جاوا در محدوده 0.8 تا 1.3 همان برنامه در C++ کارایی داشته باشد که 1.3 آن مربوط به بخش واسط کاربری و سرعت 0.8 آن مربوط به بسته تخلیه حافظه می‌شده که هیچ الگوریتمی نتوانست از الگوریتم Garbage Collector جاوا پیشی بگیرد. همچنین سال ۱۹۹۹ در مقاله‌ای آقای Lutz Prechelt به این مسئله را ثابت کردند که تجربه برنامه‌نویسی که برنامه‌ای را می‌نویسد از انتخاب زبانی که برنامه برروی آن نوشته می‌شود در کارایی تأثیر بیشتری دارد و این بدان معناست که کارایی یک برنامه را برنامه‌نویس مشخص می‌کند و نه زبان برنامه‌نویسی(ایشان در همان مقاله از زبان جاوا استفاده نمودند تا ذهنیت بد را از بین ببرند)

همچنین در صنعت نرم‌افزار هزینه اصلی مربوط به ساخت نرم‌افزار است و نه تهیه سخت‌افزار برای دستیابی به سرعت بیشتر.

حذف اشاره‌گرها در جاوا به دلیل مشکلاتی بوده که آنها در طول تاریخشان بوجود آورده‌اند، اگرچه این موارد در برنامه‌های سیستمی لازم به‌نظر می‌رسد ولی در محیط‌های تحت‌وب که بستر اصلی جاوا هستند می‌توانند اثراتی به مراتب شدیدتر نسبت به آنچه در برنامه‌های سیستمی دارند داشته باشند و باعث می‌شود که توجه برنامه‌نویسان از مسائلی چون کارایی، قابلیت اطمینان و مقیاس‌پذیری برنامه به تنظیم اشاره‌گرها معطوف گردد.

وجود وراثت چندگانه در زبانی مانند C++، باعث ایجاد مشکلات اساسی می‌گردید که اکثر برنامه‌نویسان C++ از آن دوری می‌کرده و هنوز هم می‌کنند. ولی قابلیت چندریخته شدن یک کلاس از لحاظ شی گرایی بسیار مهم بوده و بنابراین توجیهی برای وجود وراثت چندگانه را فراهم می‌نمود. در جاوا با وارد شدن مفهومی به نام واسط برنامه‌سازی (Interface)، دیگر نیازی به وجود وراثت چندگانه احساس نشد و بنابراین از زبان جاوا حذف گردید. در حال حاضر اکثر طراحان برنامه‌ها حتی به این نتیجه رسیده‌اند که وراثت تکی هم باعث ایجاد مشکل بوده و تا آنجایی که می‌شود باید از Composition استفاده نمود و در تمامی کتاب‌های طراحی که از سال ۲۰۰۰ به این طرف چاپ شده به آن اشاره نموده‌اند.

از ابتدای بوجود آمدن جاوا، کتابخانه JNI - Java Native Interface در آن وجود داشته که قابلیت فراخوانی و دستکاری برنامه‌هایی در C++ و ... را می‌داده که از نمونه‌های آن می‌توان به Jtwain که یک بسته‌ایست که از کتابخانه‌های ویندوز برای اسکن عکس استفاده می‌کند، یا SWT که یک بسته نرم‌افزاریست که از کتابخانه‌های ویندوز و لینوکس (برحسب سیستم‌عامل) برای ساخت واسط کاربری (UI) استفاده می‌کند، نام برد.

بسیاری از موارد یاد شده به عنوان ایرادات به جاوا به عنوان ایرادات به طراحی زبان‌های سطح بالا هستند و نه جاوا.

یک اشتباه متداول

برخی مردم به علت شباهت اسمی، جاوا و جاوااسکریپت را با هم اشتباه می‌گیرند. در حالیکه این دو زبان گرچه در ظاهر و کلمات شبیهند ولی بطور ساختاری با یکدیگر متفاوتند. جاوا اسکریپت محصول شرکت نت اسکیپ است.جاوا برای اجرا باید به زبان ماشین مجازی ترجمه شود اما جاوااسکریپت زبانی است که معمولا در صفحات وب نوشته می شود و توسط مرورگر تفسیر می گردد.در جاوا متغیرها همگی بر اساس نوعشان معرفی می شوند اما در جاواسکریپت نوع متغیرها به صورت ضمنی مشخص می شود.