

هرچند سال یکبار یک اتفاق بزرگ در جهان برنامه‌نویسی، مسیر برنامه‌نویسی را تغییر می‌دهد. زبان برنامه‌نویسی روبی، یکی از این اتفاق‌های بزرگ است.

روبی<sup>۱</sup> یک زبان دینامیک، همه‌منظوره، سطح بالا و شیء‌گرا<sup>۲</sup> است که دستورالعمل‌های آن از پرل<sup>۳</sup> و دیگر ویژگی‌های آن از اسمال‌تاک<sup>۴</sup> الهام گرفته‌شده‌است. روبی همچنین از ساختارگرایی و انکاس پشتیبانی می‌کند. علاوه بر استفاده از سیستم دینامیک، مدیریت حافظه در روبی کاملاً خودکار است. بنابراین روبی از بسیاری از جهات شبیه پیتون، پرل و لیسپ است.

- پیدایش**

روبی در اواسط دهه ۱۹۹۰ در ژاپن و توسط یوکی هیرو ماتسوموتو که به متر<sup>۵</sup> نیز معروف است پا به عرصه وجود گذاشت. ماتسوموتو در مورد نامگذاری روبی می‌گوید: «روبی که در لغت به‌معنای یاقوت است سنگ ماه تولد یکی از همکاران من است.» روبی سنگ متولدین ماه جولای است. ماتسوموتو که خود به زبان‌های لیسپ، پرل، اسمال‌تاک و پایتون <sup>۶</sup> مسلط است، در مورد دلیل پیدایش روبی می‌گوید: «من به دنبال یک زبان اسکرپت‌نویسی می‌گشتم که از پرل قدرتمندتر و از پایتون شیء‌گراتر باشد. از آنجاکه چنین زبانی وجود نداشت، تصمیم گرفتم زبان ایده‌آل خود را پدید آورم.» به این ترتیب روبی عضو جدیدی از خانواده پرل و پایتون به‌شمار می‌رود.

- امکانات**

روبی دارای کتابخانه‌های گسترده‌ای است که به همه نیازهای برنامه‌نویسان به‌خوبی پاسخ می‌دهد. روبی همچنین از کلاس‌های FTP، پشتیبانی از CGI، پارسرهای XML، کتابخانه‌های پایگاه‌های داده، GTK و Qt و حتی یک کتابخانه BTree برخوردار است. کتابخانه‌های روبی قابل اتصال به کتابخانه‌های

پایتون هستند و این مساله پایه قویتری را برای برنامه‌نویسی روبی فراهم می‌کند. روبی توسط اکثر سیستم عامل‌ها از جمله لینوکس، Mac OS X، مایکروسافت ویندوز، ویندوز CE، و یونیکس پشتیبانی می‌شود. اجرای روبی روی یونیکس بهترین کارایی را دارد.

متغیرهای روبی در پنج سطح جهانی<sup>۷</sup>، محلی، کلاس، بلوک و نمونه<sup>۸</sup>

# روبی؛ زبان برنامه‌نویسی مولد

**سرعت توسعه‌کد**

سرعت کدنویسی برنامه‌نویسان در طی سال‌های گذشته رشد چشمگیری داشته است. هرچند آمارهای رسمی در این زمینه موجود نیست، اما به‌طور متوسط هر پنج سال یک‌بار سرعت‌آماده‌سازی نرم‌افزارها دو برابر می‌شود. از این گذشته اکنون منحنی تولید نرم‌افزارهای قابل نگهداری، رشد خطی دارد. بخش اعظم این سرعت مربوط به ابزارهایی است که برنامه‌نویسان استفاده می‌کنند و در قلب این ابزارها زبان برنامه‌نویسی قرار دارد. زبان‌های برنامه‌نویسی در اصل مشابه یکدیگرند. در مقایسه زبان‌های گوناگون، شباهت‌های کلی زبان‌ها از تفاوت‌های آنها بیشتر است. با این حال، تفاوت‌های موجود در میان زبان‌های مختلف اثر قابل‌ملاحظه‌ای بر حل مسائل، یافتن راه حل و برهم‌کنش انسان و رایانه دارند. به‌عنوان مثال، در زبان لیسپ، زبانی که زیبایی فوق‌العاده و سادگی در مفاهیم را در هم آمیخته است، بخش عمده توانایی‌های زبان در خدمت رایانه قرار گرفته است و نه در خدمت انسان. زبانی که تفسیر آن ساده است، الزاما زبان ساده‌ای برای برنامه‌نویسی نیست و این مساله‌ای است‌که اغلب برنامه‌نویسان آن را تجربه کرده‌اند. شگفت‌انگیزتر ین ویژگی لیسپ این است‌که نشان می‌دهد چگونه می‌توان با کمترین تعداد قوانین یک زبان رایانه‌ای قدرتمند خلق کرد. در مقابل، روبی چنان هنرمندانه نوشته شده است که کد آن مانند شعر به‌نظر می‌رسد.

یک مطالعه بسیار جالب در زمینه سرعت توسعه کد نشان می‌دهد که سرعت توسعه کد رابطه مستقیم با تعداد خطوط کد نوشته شده دارد. تعداد خطوط کد در پروژه‌های نوشته‌شده با ++C و جاوا تقریبا دوبرابر تعداد خطوط پروژه پایتون یا پرل است. به این ترتیب توسعه کد در ++C و جاوا دو برابر زمان می‌برد. نوشتن کد با روبی سرعت توسعه‌کد را به مراتب بیشتر از پرل و پایتون افزایش می‌دهد و در این زمینه از آنها پیشی می‌گیرد. دستورات روبی بسیار ساده و قابل درک هستند و کد روبی به‌زبان طبیعی بسیار نزدیک است. به‌عنوان مثال، در روبی از نقطه‌ویرگول (سمی‌کالن) استفاده نمی‌شود. عد

وجود عواملی نظیر نقطه‌ویرگول که در برنامه‌نویسی وقفه ایجاد می‌کند، سبب می‌شودکه برنامه‌نویسی یک‌روندساده‌تر و روان‌تر پیداکنند. درحقیقت سادگی دستور زبان روبی و سهولت تعریف کردن کلاس‌های جدید سبب می‌شود که نوشتن کلاس‌ها ساده‌شده و از آنها در تست کد در حد کلاس استفاده شود. در نتیجه طراحی‌شیء‌گرا بسیار سریع و ساده می‌شود. به این ترتیب برنامه‌نویس درصدد جستجوی راه‌حل‌های دیگر بر نمی‌آید.

- ویژگی‌های انحصاری**

برخی از ویژگی‌های روبی، در زبان‌های دیگر وجود ندارند. روبی از ساختار closure پشتیبانی می‌کند. در این ساختار مسولیت هر بخش از کد بر عهده شیء مناسب قرار می‌گیرد و کد حاصل کد خواناتری خواهد بود. نوشتن حلقه‌های for و while در روبی نادر است. همچنین پیاده‌سازی ساختارهایی نظیر "each" که احتمال رخداد خطا در آن کمتر است، به‌سادگی امکان‌پذیر است. در پروژه‌های پیچیده مساله قابلیت نگهداری کد، به یکی از مسایل مورد بحث تبدیل می‌شود. طراحی شیء‌گرای مناسب به برنامه‌ساختار مناسب داده

## برای درک بهتر

به‌عنوان مثال دریک برنامه برای نگه‌داشتن «نام‌پدر» می‌توان از متغیری به نام Aاستفاده کرد. یا با کمی تخفیف از حرف F (حرف اول کلمه Father) استفاده کرد!اما این‌یک اشکال بزرگ دارد. اگر شخص دیگری برنامه شما را ببیند متوجه نمی‌شودکه در منطق برنامه شما، متغیر a یا f برای نگهداری چه مفهومی در نظر گرفته شده است. خود برنامه‌نویس نیز ممکن است پس از گذشت مدت زمان طولانی، با رجوع به برنامه خود، به خاطر نیاورد که a یا f را به چه منظوری تعریف کرده‌است. بنابراین بهتر است برای متغیرها نام مناسب و مربوط به محتوای آن انتخاب کنیم. مثلا در مثال قبل، انتخاب شناسه father name در هر زمان برای هر کسی گویای این خواهد بود که این متغیر برای نگهداری «نام‌پدر» در نظر گرفته‌شده‌است.

برخی برنامه‌نویسان با دانستن این مساله باز ترجیح می‌دهند از نام‌های کوتاه استفاده کنند و دلیل آنها برای این کار نیز سادگی و سرعت تایپ و جلوگیری از خط در تایپ نام‌های طولانی است. این دلیل در ویراستارهای قدیمی تا حدی قانع‌کننده است. اما در ویراستارهای



و کد برنامه را منطقی و واضح می‌کند. آنچه‌که مسلم است، این است که همه زبان‌های شیء‌گرا تا حدی از شیء‌گرا پشتیبانی می‌کنند. به‌عنوان مثال زبان ++C بیشترین پشتیبانی را از شیء‌گرایی به‌عمل می‌آورد در حالی‌که جاوا در این زمینه اندکی ضعیف‌تر عمل می‌کند. اما روبی از نظر سادگی در نحوه استفاده و قابل پیش‌بینی بودن از همه زبان‌های دیگر پیشی گرفته است.

- محدودیت‌ها**

مانند همه زبان‌های دیگر، به‌روبی نیز انتقاداتی وارد است. سرعت اجرای کد روبی در مقایسه با بسیاری از زبان‌هایی که کد را کامپایل می‌کنند، پایین است. سرعت روبی با سرعت زبان‌هایی که از مفسر استفاده می‌کنند و دیگر زبان‌های اسکرپت‌نویسی مانند پرل و پایتون قابل مقایسه است. تعریف متغیرها در روبی بسیار ساده است و تنها با تخصیص یک مقدار به متغیر می‌توان یک متغیر جدید تعریف کرد.

اشتباه‌های تایپی می‌توانند سبب معرفی متغیرهای تازه شده و منجر به بروز رفتارهای غیرمنتظره شوند. به‌دلیل دینامیک بودن انواع داده در روبی،

جدید اغلب امکانی به نام Autocomplete وجود دارد که به برنامه‌ساز امکان می‌دهد با کمترین میزان تایپ و تنها با وارد کردن حروف اول شناسه، گزینه مورد نظر خود را از لیست ظاهر شده به راحتی انتخاب کند.

این امکان، هم سرعت را افزایش می‌دهد هم اشتباه تایپ را از بین می‌برد و هم امکان نامگذاری‌های بلندتر را بدون اشکال می‌سازد.

دلیل دیگری نیز برای برنامه‌نویسان جهت استفاده از نام‌های کوتاه و بی‌مفهوم وجود دارد و آن بالا بردن امنیت کد است. به عبارتی آنها بر این اعتقادند که ما می‌دانیم هر چیز را برای چه استفاده کرده‌ایم و یا در جایی برای خودمان ثبت می‌کنیم. اما می‌خواهیم اگر تصادفا این کد دست بیگانه افتاد، در درک آن دچار مشکل شود و از آن سردرنیاورد. این توجیه تا حدی قابل قبول است. حتی امروزه نرم‌افزارهایی وجود دارندکه کد برنامه‌های متن باز مثل جاوا اسکرپت یا Php را به شکلی درمی‌آوردنکه درک آن تا حد امکان مشکل شود و به این ترتیب امنیت آن را در برابر سرتقت بالا می‌برند.

- مستندسازی**

یکی از مهم‌ترین کارهایی که یک برنامه‌نویس باید در برنامه‌نویسی خود رعایت کند مفهوم مستندسازی یا Self Documenting است. بعد از این که این مفهوم را توضیح دادیم، همیشه آن را به خاطر داشته

اگر به متغیری مقداری از نوع داده‌ای ناهماهنگ نسبت داده شود، خطاهای زمان اجرا رخ خواهند داد.

در برخی از زبان‌ها تبدیل انواع به‌طور خودکار انجام می‌شود، یا اشتباه در مقداردهی توسط مفسر زبان به برنامه‌نویس اعلام می‌شود اما روبی فاقد مکانیسمی برای رفع این مشکل است. روبی از رشته‌های چندبایتی و نیز از یونیکد پشتیبانی نمی‌کند. همچنین در حال حاضر سیستم روبی فاقد یک تعریف معین و استاندارد است. ماتسوموتو امیدوار است که در آینده‌ای نه‌چندان دور مستندات مربوط به یک تعریف مشخص از روبی تکمیل شوند و روبی استاندارد ایزو دریافت‌کند.

- قالب‌کاری**

قالب‌کاری روبی، روبی‌روی‌ریل<sup>۱۰</sup> یا به اختصار ROR نامیده می‌شود. ROR یک قالب کد باز تحت وب است که در میان توسعه‌دهندگان کدهای تحت وب طرفداران زیادی دارد. ROR برای توسعه پروژه‌های کوچک و مبتنی بر سیستم کلاینت–سرور مناسب است. با استفاده از ROR برنامه‌نویس تنها باید برای موارد خاصی که در قالب پیش‌بینی نشده‌است کد بنویسد. به‌عنوان مثال اگر در برنامه کلاسی به‌نام "فروش" وجود داشته باشد، در پایگاه داده‌ها جدولی به همین نام ایجاد می‌شود. برنامه‌نویس تنها زمانی نیاز به کدنویسی مجدد دارد که نام جدول به کالاهای فروخته‌شده<sup>۱۱</sup> تغییر کند. به این ترتیب استفاده از ROR حجم کدنویسی را کاهش می‌دهد. از دیگر مزایای ROR می‌توان به‌وجود کتابخانه‌های گسترده جاوا اسکرپت و خروجی به فرم HTML و XML اشاره کرد. ROR در سال ۲۰۰۷ در در سازمان ثبت اختراعات آمریکا موسوم به USPTO به ثبت رسیده است.

**پی‌نوشت**

- Ruby
- Object Oriented
- Perl
- Smalltalk
- Matz
- Python
- Global
- Instance
- Matz's Ruby Interpreter
- Ruby on Rails

می‌رود یک متغیر در جایی داشته باشد، کاری که یک دستور یا یک تابع انجام می‌دهد و هر آنچه که فکرش را می‌کنید در برنامه اضافه کنید تا هم خودتان و هم هر کس دیگر که برنامه را می‌خواندکل منطق آن را متوجه شود. به ساختار یک نمونه توضیحات گذاری در دلفی توجه‌کنید:

```
//a function for Input
Function f ( )
Begin
If p = = 1 {Input is ready} then
...
End
```

در مثال فوق عبارت a function for input و هر آنچه بعد از آن تا پایان خط باید همچنین نوشته‌های داخل { یعنی Input is ready توسط کامپایلر نادیده گرفته می‌شود و می‌تواند حتی به زبان فارسی وجوددارد. در ++C استفاده از /\*\*...\*/و در دلفی { ... }

بنابراین شما مشخص کرده‌ایدکه تابع f ( ) برای گرفتن ورودی است ووقتی متغیر P برابریک‌شودیعنی ورودی آماده است. به کدی که در آن توضیحات گذاری به میزان کافی رعایت شده باشد یک‌کد مستندسازی شده یا خودمستندسازی شده گفته می‌شود و این نوع امتیاز بالایی خواهد داشت.