KEY CONSIDERATIONS WHEN SELECTING THE RIGHT LANGUAGE(S) FOR YOUR WEB APPLICATIONS

A REVIEW OF JAVA, PYTHON, RUBY, GO, NODE.JS AND PHP

ActiveState

WEB LANGUAGE GUIDE

ActiveState

INTRODUCTION

These days every company is a technology company. Every company, whether in finance, healthcare, manufacturing, transportation or any other industry, is moving to a digital-first strategy in order to deliver great customer experiences and stay ahead of the competition.

As part of this transformation, companies are recognizing the need to put technology choices at the core of their culture by giving developers more of a say in which tools are most suitable for the job. When it comes to web applications, developers are calling on enterprises to embrace open source languages, which allow for greater flexibility and collaboration.

However, selecting the right language for your project can be a daunting task. The number of libraries, frameworks, tools and services available for web applications today can be overwhelming. The sheer number of moving parts involved, from cloud hosting services, microservices, containers, NoSQL databases, MVC frameworks, REST APIs and the many other dizzying array of acronyms, can make wading through the sea of options difficult for even the most tech-savvy decision makers.

Choosing the right language for your project comes down to more than just the language syntax—there are a number of operational variables to take into consideration:

Developer Familiarity: How familiar are your current developers with this language? Some developers love to learn new technologies, but recognize that their efficiency will be reduced while they become familiar with them and learn the best practices. Others may resist moving away from your tried and true enterprise languages.

Availability of Talent: New technologies might offer features and innovations that aren't available with more established languages, but you also may not have a deep talent pool to hire from. Some open source languages are easier to learn than others and offer growth opportunities for your staff, on top of the innovations you want to acquire. Consider both the breadth of the available talent, as well as the depth to get an understanding of how this will impact both your timeline and costs.

Language Properties: Understand your project requirements to make a determination about which language has the innate properties that make it the best fit for the task at hand. Object-oriented languages offer a powerful representation tool for complex systems, but concurrency is increasingly important in scalable systems. Dynamic typing contributes to rapid development, but may come at the cost of execution speed, may require deeper test coverage, and it will be more difficult to improve code quality using static analysis tools. All of these create trade-offs when selecting the best language (or combination of languages) for your project or application.

Language Ecosystem: Availability of suitable frameworks, libraries and community support can have a dramatic impact on your project's success. Understanding what tools are available to manage dependencies, and what the stability and maturity of 3rd party libraries are within a given language ecosystem is critical to your decision.

WEB LANGUAGE GUIDE

ActiveState

The table below compares the most popular languages for modern web development and can help in selecting the best languages for the job.

	Python	Ruby+Rails	Node.js	Go	Java	PHP
Core Strengths/ Language Philosophy	Rapid prototyping, Object-oriented, Accessible, Embedded parsing	Convention over configuration, DRY, Object-oriented, Embedded parsing	Rapid development, Asynchronous, 'Javascript everywhere', Embedded parsing	Concurrency, Clean+concise code, Performant	JVM performance, Object-oriented, Cross-platform	Web-server integrated, Embedded parsing
Ideal Use Cases	SaaS, Machine learning, Data science, Big data, Rapid development	SaaS, API services, End user apps, App server, Rapid development	SaaS, API services, End user apps, Rapid development	Microservices, Big data, App server/APIs, Streaming metrics, Data science	App server, Big data, API service	End user apps, App server, SaaS
Developer Talent Pool	Deep	Deep	Deep and growing	Growing	Deep	Mature but shrinking
TIOBE Index Rank[1]	#5	#11	n/a[2]	#18	#1	#6
Learning Curve	Highly accessible, easy to train, commonly taught in schools	Steeper learning curve, but productive quickly	Easy to learn, familiar for FE devs	Steeper learning curve, but less so than functional languages (Scala, Elixir)	Commonly taught in schools, steeper curve	Easy to pick up but code maintenance difficult
Ecosystem	Mature, robust package ecosystem with most areas covered	Mature, robust package ecosystem with most areas covered	Exploding growth with packages for almost any use case	Growing package ecosystem, not everything is covered natively yet	Extremely mature	Mature, but lacks some coverage on modern tool integrations
Performance	Reasonable performance	Reasonable, can become laggy under load	Reasonable	High performance, built-in concurrency	High performance	Can be laggy under load
Popular Web Framework	Django	Rails	Express	Echo/Gin	Spring MVC	Zend/Drupal
Semantic Properties	Dynamic	Dynamic	Dynamic	Static	Static	Dynamic
Dependency Management	PIP provides robust management, updates and versioning	Ruby Gems provides robust management, but versioning can become complex	npm provides robust management, updates and versioning	dep is emerging as the integrated solution but not standard yet	Maven and others provide robust package management	Various
Deployment	Many deployment tools/integrations	Many deployment tools/integrations	Wide cloud integration, easy to deploy	Compiled executable, the simplest possible, DevOps dream	JVM ubiquity	Easy, built into Apache, wide cloud integration
Other Language Bindings	Highly flexible and has easy C bindings	Highly flexible, with many C bindings	Flexible	Extremely flexible with seamless C bindings	More difficult to integrate with other languages	More difficult to integrate

[1] As of April 2017, taken from https://www.tiobe.com/tiobe-index/. TIOBE Index is an "indicator of the popularity of programming languages". Further definition can be found here: https://www.tiobe.com/tiobe-index/programming-languages-definition/.

[2] Node.js is not ranked separately in TIOBE, but Javascript is ranked #8, where Node.js is a sub-set of this rank.

LANGUAGE SUMMARIES

Ultimately, each language has its own ecosystem of tools and frameworks, but with the increasing popularity of microservice architectures, it's becoming common to mix them together to create hybrid applications that play to the strengths of each component of the system. When making the decision about what language to use for your next project, there are a number of key factors that you want to consider depending on your use-case.

Python

Python is an extremely accessible and flexible language that is arguably the dominant language in the big data, machine learning and scientific computing communities. Prized for its ability to manipulate large data sets and for its visualization tools, combined with a robust web framework like Django, it is an excellent choice for many enterprises as they start to mine their data for advanced insights.

Python offers clean syntax, and is unique for its use of whitespace to denote blocks of code. It supports object-oriented programming, is dynamically typed, and has a comprehensive ecosystem of packages that are well supported by an active community.

Well-known sites powered by Python: Pinterest, Instagram, Disqus, Bitbucket, Eventbrite

Ruby on Rails

The Rails web framework has become almost synonymous with Ruby. Popularized by Basecamp (then 37signals) and developed by David Heinemeier Hansson, it has been driven to popularity by its opinionated design, which allows developers to rapidly build applications by adhering to the strict conventions laid out by Rails. For both typical CRUD (Create, Read, Update, Delete) applications or API services, Rails is a popular choice due to its widespread adoption.

Some early criticisms of Ruby were that Rails applications had scaling problems in large production installations, but many high profile websites continue to use Ruby on Rails in these conditions. Later versions of Rails support API-only services, making it an increasingly popular choice for REST-based service applications.

Well-known Rails sites: Twitter, Kickstarter, Soundcloud, Bloomberg, Crunchbase, Shopify, Basecamp

Node.js

The "new kid on the block"—Node.js offers a unique single-threaded asynchronous architecture that makes development both approachable and fast. Because it's just Javascript on the server, it's already familiar to the multitudes of front-end programmers out there who are comfortable with Javascript. Node's extreme popularity has led to an explosion of package creation, and the popular npm package manager makes accessing this thriving ecosystem a breeze. Already, Node.js is making its mark on the enterprise with a number of large scale sites being built with Node.

Node.js is among the fastest growing language ecosystems at the moment, with the number of new packages being added skyrocketing. While many of the nascent extensions aren't necessarily battle-tested or production ready, they are actively maintained and being put into real-world use by a rapidly increasing number of organizations.

Well-known Node.js sites: Paypal, eBay, LinkedIn, Netflix, Uber, Trello, Medium

Go

Developed by Google, Go is a modern advancement on compiled languages. It combines the system-level power and performance of a compiled language like C with the safety, convenience and rapid development of an interpreted language. It is especially powerful by making concurrency easy through the use of 'goroutines'. This makes Go an especially good choice for back end systems that need to scale, and combined with core language design features, make it the top choice for microservices development.

Go is a language that is designed to be 'read' and its excellent code readability, strict package design requirements, and the Interface language component make it one of the most easily maintainable languages available. With a compatibility promise from the Go team around not introducing breaking changes to the language spec, long-term maintenance and stability make Go a top choice for enterprises.

Popular sites using Go in their stack: YouTube, Dropbox, Heroku, Bitly, Google, Cloudflare

Java

Java has been the reigning and de facto king of server-side development for almost 20 years, and with ongoing optimizations to the JVM, its performance rivals that of compiled languages like Go or C/C++. With a large package ecosystem and a deep pool of available talent, it is often considered the 'safe' choice.

However, unlike many of the other dynamic languages like Ruby or PHP, Java does not support embedded parsing within web pages, and must do all of the page rendering itself. Any product or service that has a public-facing component will need another solution to implement that front-end, such as Javascript. Additionally, Java is a famously heavy-weight solution, and the age and maturity of its many, many frameworks can result in large, complex code-bases that in other languages might be smaller and more amenable to rapidly scalable systems.

Java developers tend to be very experienced and will have little difficulty learning a new language. Despite not being an object-oriented language, Go may seem familiar to many Java programmers since it is a compiled language that excels in many of the same use cases as Java.

PHP

PHP is the granddaddy of web development languages and for most of the early web would have been one of the entry-points into web development. Because it is often bundled with popular web servers like Apache, its availability on web hosts is nearly universal. PHP is embedded directly into web pages and parsed server-side, so it is easy to integrate into existing static sites. However, because it is tied closely with client-side code, large scale applications can be difficult to maintain and scale.

Popular sites using PHP: Facebook, Wikipedia, Wordpress, Flickr

RECOMMENDATIONS

Which language is the right fit for your organization? It truly depends on the specific use case, as all these languages are equally capable of delivering on almost any requirement. However, you need to prioritize the key goals for your project to choose the language that is the best fit for your particular use case. In today's environment though, having a single language solution is unlikely to keep your organization competitive or meet the diverse needs of a rapidly changing marketplace. Modern languages provide benefits that range from rapid development cycles, dynamic package ecosystems, and a burgeoning workforce who are current on all the latest technologies.

Some things to keep in mind when making your decision:

Talent Pool - The talent pool in popular and growing languages will be more accessible and likely less-expensive when compared to mature development pools.

User-Facing vs. APIs - For purely server-side APIs, languages like Go and Java excel. Whereas for user-facing applications, languages like Python and Ruby have a large number of frameworks and tools that already exist for most common business use cases—including the required user-facing JS/HTML in many cases—and their design lends them to rapid development and quick iteration in response to A/B testing.

Existing Knowledge - Leveraging the existing skillsets of your developers is important—languages like Node.js can be attractive since any JS dev will be comfortable using it without much additional training required.

Performance/Scalability - As the complexity of web apps increases over time, considerations about performance and scalability become critical and a microservices model starts to become attractive. In this case, consider using best-of-breed solutions for each component of the system (see Figure 1)—which may mean Go for performance-critical applications on the back-end, Ruby for API-based services, and Node.js or Python for any user-facing services.

Integration - Any new project will have to interface with legacy systems, and as organizations move towards hybrid microservice models, this approach allows new projects to adopt the latest technology, while mission-critical legacy systems can remain in operation uninterrupted. Thankfully, modern dynamic languages all have excellent bindings, support for message brokers and other protocols. The limiting factor will almost certainly be your existing legacy system's ability to interface with more modern tools.

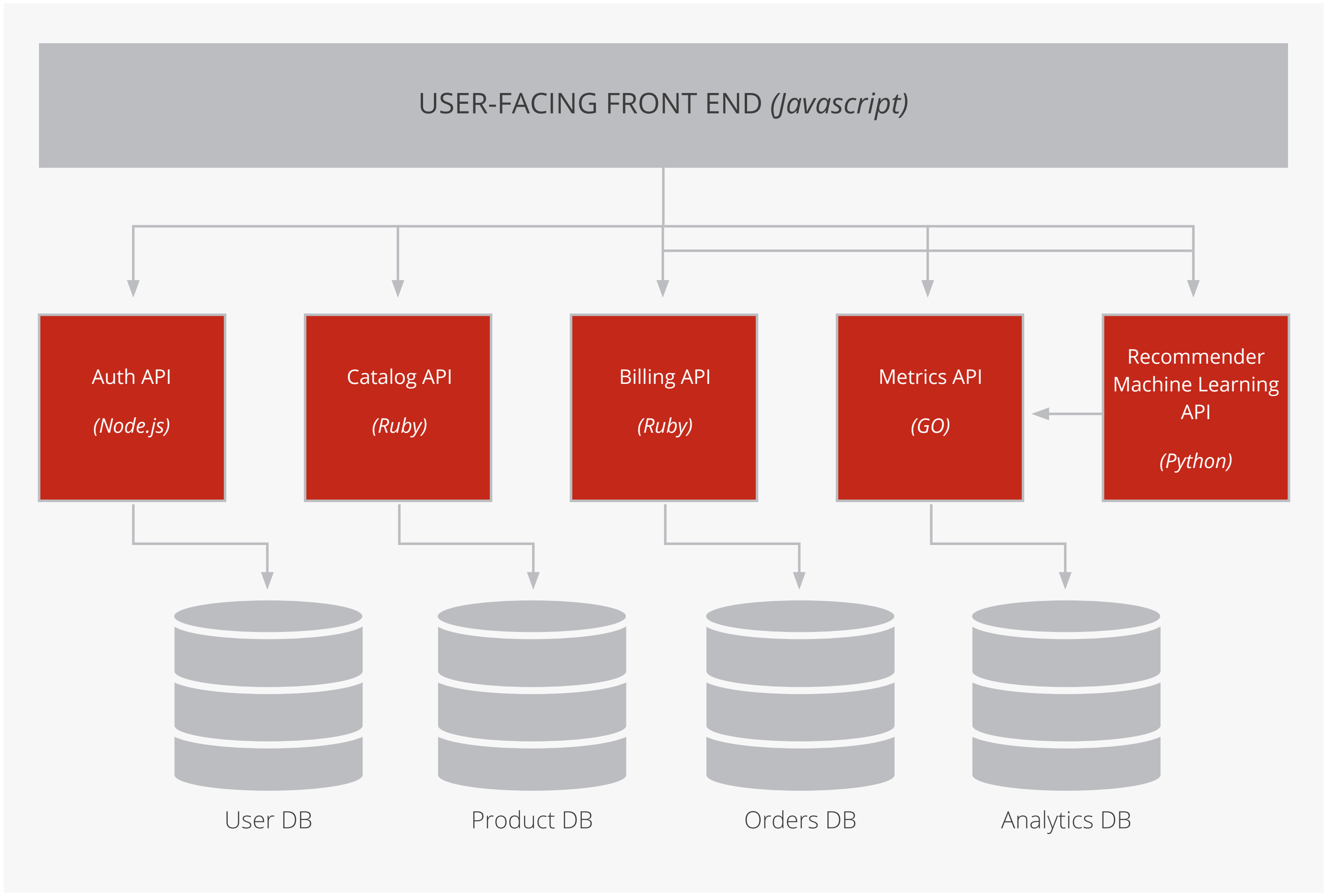

Figure 1. Sample Hybrid Microservice Web App

Modern web applications are increasingly complex and their interactions with different business units within an enterprise make the ability to have a single monolithic solution to rapidly changing needs no longer a viable solution. Adopting the right tool for the job can make your business more dynamic to changing customer needs. Leaving behind the idea that, "we're a Java shop" and embracing the hybrid ecosystem of modern languages can let you stay ahead of the competition.

ACTIVESTATE: GET STARTED WITH OPEN SOURCE LANGUAGES

Choosing the right languages is crucial to your digital strategy success, but brings with it a few additional considerations. Managing the hundreds of options with 3rd-party open source packages developers use along with each language and framework can be time-consuming, low-value work. Getting support from the community can be risky, as you may not want to openly discuss the details of your applications. Security is another top concern, as downloading open source packages exposes your company to risks—both from a security vulnerability standpoint and license perspective.

ActiveState has you covered with pre-bundled language distributions for Python, Go, Ruby and Node.js*, and our Komodo IDE multi-language IDE for web development. With the most popular packages already included, your team can spend more time on development and getting your applications to market faster. ActiveState distributions provide the latest secure versions of packages, along with timely security updates for critical issues. Plus, you get the support you need for technical issues—from the code on your developers' desktops to production—no matter where your applications live. With ActiveState, you can get the most productivity out of your choice of web language, with the least amount of risk.

Start developing for free with Community Edition, and learn more about our commercial options for use in production: www.activestate.com

*ActiveGo, ActiveRuby and ActiveNode scheduled for general availability in 2017.

ActiveState Software Inc.
business-solutions@activestate.com

Phone: **+1.778.786.1100**
Fax: **+1.778.786.1133**

Toll-free in North America:
1.866.631.4581

ActiveState®

ABOUT ACTIVESTATE

ActiveState, the Open Source Languages Company, believes that enterprises gain a competitive advantage when they are able to quickly create, deploy, and efficiently manage software solutions that immediately create business value, but they face many challenges that prevent them from doing so. The Company is uniquely positioned to help address these challenges through our experience with enterprises, people and technology. ActiveState is proven for the enterprise: More than two million developers and 97% of Fortune-1000 companies use ActiveState's end-to-end solutions to develop, distribute, and manage their software applications. Global customers like Bank of America, CA, Cisco, HP, Lockheed Martin and Siemens trust ActiveState to save time, save money, minimize risk, ensure compliance, and reduce time to market.