**Linguagens de programação**

**Resumo**

**Questão**

É necessário escolher a linguagem de programação para o sistema de front-end adequada para web e uma linguagem de back-end adequada para servidor.

**Decisão**

Estamos escolhendo o CSS para o front-end.

Estamos escolhendo PHP para o back-end.

**Status**

Decidido. Estamos abertos a novas alternativas à medida que surgem.

**Detalhes**

**Suposições**

Queremos criar software para web que sejam modernos, rápidos, confiáveis, responsivos etc.

Os aplicativos de front-end evoluem rapidamente:

* Queremos garantir desenvolvimentos, implantações, iterações rápidas e fáceis, etc.
* Valorizamos a comprovação, como a segurança de tipo, e estamos bem trabalhando um pouco mais para alcançá-la.

Os aplicativos de back-end são superiores ao típico:

* Metas acima do normal para qualidade, especialmente provabilidade, confiabilidade, segurança, etc.
* Metas acima do normal para quase tempo real, ou seja, não queremos pausas devido à coleta de lixo da máquina virtual.
* Objetivos acima do normal para programação funcional, especialmente para paralelização, processamento de vários núcleos e segurança de memória.

Aceitamos velocidades de tempo de compilação mais baixas em favor da segurança de tempo de compilação e velocidades de tempo de execução.

**Restrições**

Temos uma forte restrição de conhecimento da linguagem de programação.

**Posições**

Nós consideramos estas linguagem:

* C
* Java
* JavaScript

**Argumento**

Resumo por linguagem:

* C: rejeitado devido à baixa segurança;
* Java: excelente tempo de execução; excelente ecossistema; experiência de desenvolvedor abaixo da média.
* JavaScript: a linguagem mais popular de todos os tempos; ecossistema mais difundido.
* PHP:

Decidimos que as VMs têm um conjunto de compensações de que não precisamos no momento, como complexidade adicional que fornece recursos de tempo de execução.

**Implicações**

Os desenvolvedores de front-end precisarão aprender TypeScript. Esta é provavelmente uma curva de aprendizado fácil se a experiência principal do desenvolvedor for usar JavaScript.

Os desenvolvedores de back-end precisarão aprender o Rust. Esta é provavelmente uma curva de aprendizado moderada se a experiência principal do desenvolvedor estiver usando C / C ++, e uma curva de aprendizado difícil se a experiência principal do desenvolvedor estiver usando Java, Python, Ruby ou linguagens gerenciadas por memória semelhantes.

TypeScript e Rust são relativamente novos. Isso significa que muitas ferramentas ainda não possuem documentação para esses idiomas. Por exemplo, o pipeline devops precisará ser configurado para esses idiomas e, até agora, nenhuma das ferramentas devops que estamos avaliando tem exemplos padrão para esses idiomas.

Os tempos de compilação para TypeScript e Rust são bem lentos. Parte disso pode ser devido à novidade das línguas. Podemos querer ver como mitigar tempos de compilação lentos, como por compilação sob demanda, simultaneidade de compilação, etc.

O suporte IDE para essas linguagens ainda não é onipresente e ainda não é de primeira classe. Por exemplo, a JetBrains vende o PyCharm IDE para suporte de primeira classe para Python, mas não vende um IDE com suporte de primeira classe para Rust; em vez disso, o JetBrains pode usar um plug-in Rust que fornece talvez 80% do suporte à linguagem Rust em relação ao suporte à linguagem Python.

**Relacionados**

**Decisões relacionadas**

Nosso objetivo é fazer escolhas de ecossistemas que se alinhem a esses idiomas.

Por exemplo, queremos escolher um IDE que tenha boas capacidades para essas linguagens.

Por exemplo, para nosso framework web front-end, temos mais probabilidade de decidir sobre um framework que tende a apontar para o TypeScript (por exemplo, Vue) do que um framework que tende a apontar para JavaScript simples (por exemplo, React).

**Requisitos relacionados**

Todo o nosso conjunto de ferramentas deve oferecer suporte a esses idiomas.

**Artefatos relacionados**

Esperamos que possamos exportar alguns segredos para variáveis ​​de ambiente.

**Princípios relacionados**

Meça duas vezes, construa uma vez. Estamos priorizando alguma segurança em relação a alguma velocidade.

O tempo de execução é mais valioso do que o tempo de compilação. Estamos priorizando o uso do cliente sobre o uso do desenvolvedor.