Sencha

Uma Empresa da Idera, Inc.

A Evolução da Ferramenta JavaScript

UM GUIA DO DESENVOLVEDOR MODERNO

Alan R. Weiss

INTRODUÇÃO

O código-fonte de aplicações JavaScript tem sido, por tradição, difícil de compreender pelo fato de ser disseminado em arquivos JavaScript, HTML e CSS, além de eventos e dados fluindo por diversos caminhos não intuitivos. Como todos os softwares, o ambiente de desenvolvimento JavaScript inclui criadores de pacotes, gerenciadores de pacotes, sistemas de controle de versão e ferramentas de teste. Cada um desses itens exige uma certa curva de aprendizado.

As inconsistências e incompatibilidades entre navegadores exigiam historicamente que diversos ajustes e casos especiais fossem colocados no código e, muito frequentemente, corrigir um problema em um navegador gerava outro problema em outro navegador. Como resultado, as equipes de desenvolvimento lutam para criar e manter aplicativos de grande escala de alta qualidade, enquanto a demanda para o que eles fazem aumenta, especialmente no nível de aplicações corporativas em que o impacto dos negócios substituiu a pergunta “Quantas linhas de código você criou?”

Para lidar com essa complexidade, a comunidade de código aberto e as empresas criaram diversas estruturas e bibliotecas, mas essas estruturas e bibliotecas se tornaram ainda mais complicadas à medida que adicionam mais e mais recursos em uma tentativa de facilitar para o desenvolvedor. Mesmo assim, as estruturas e bibliotecas oferecem vantagens significativas para desenvolvedores e também podem organizar e até mesmo reduzir a complexidade.

Esse guia discute algumas das estruturas e bibliotecas mais populares que foram criadas para facilitar a carga da criação de códigos de interface de usuário (UI) complexos e como aplicações corporativas, especialmente com muitos dados, podem se beneficiar do uso dessas estruturas e componentes de UI para fornecer aplicações mais rapidamente, com melhor qualidade e ainda permanecer dentro do orçamento de qualquer agência de desenvolvimento.

Complexidade do desenvolvimento Web moderno

Andrew S. Tanenbaum, o inventor do Minix (um precursor do Linux, normalmente usado para criar novos chips e sistemas de computador), disse uma vez: “O mais legal sobre padrões é que você tem muitos para escolher.”
[1] Os navegadores seguiam diversos padrões, mas não todos eles, e muitos apenas eram independentes. Foi aí onde o problema começou, a tão conhecida “Guerra de navegadores”. A maneira como cada navegador exibia os dados desses sites da Web pode ser muito diferente. As incompatibilidades de navegadores ainda existem hoje e é possível afirmar que elas estão um pouco piores porque a Web agora se tornou móvel.

O desenvolvimento no mundo de hoje significa ser o mais compatível possível com o máximo de navegadores Web populares possível, incluindo dispositivos móveis e tablets.

E dispositivos móveis?

Aprender Android Java (Android) pode ser difícil se o desenvolvedor não tiver um conhecimento de Java. Para Apple iOS, o Objective C é uma combinação da linguagem de programação C e Smalltalk, que é diferente, mas não completamente estranho para desenvolvedores de C++. (No fim das contas, os conceitos voltados a objetos são semelhantes.) Mas com a vinda do (Apple) Swift e de um novo paradigma, a programação “voltada para protocolo”, o Objective C tem um futuro questionável.

Em contraste, o mundo JavaScript, por meio de técnicas como aplicativos nativos do React ou da Web progressivos, permite o desenvolvimento de aplicativos cross-plataforma que se parecem com aplicativos nativos e possuem um bom desempenho. Do ponto de vista corporativo, uma empresa pode obter diversas vantagens apenas ao usar um conjunto de ferramentas para criar aplicativos Web e móveis sofisticados.

Consternação de causas de mudanças constantes

O mundo JavaScript é particularmente rico na quantidade de funcionalidades e pacotes que estão disponíveis. A quantidade é surpreendente. A quantidade de tecnologias-chave que ajudam os desenvolvedores a criar aplicativos mais rapidamente também é grande, mas a taxa de mudança no campo causa a famosa “evasão do JavaScript”, ou apenas evasão. Por exemplo, quando o Angular mudou da versão 1 para a 2 (e novamente da 3 para a 4), as incompatibilidades exigiam um tempo de portabilidade significativo. Até que abracemos os padrões de Componentes da Web emergentes2, nem tudo terá uma intercomunicação com todas as outras coisas.

Uma coisa que pode ser dita é que investir em tecnologias antigas sem o suporte dos padrões pode ser limitador, portanto, a importância das normas de ECMA e ECMAScript, bem como a aderência a padrões de design mais ou menos comuns (a maioria da programação ainda é, até mesmo ultimamente, a manutenção de código existente em vez de criações e arquiteturas novas). Usar padrões de design normalmente utilizados, como Model-View-Controller (MVC), Model-View-Viewmodel (MVVM) e Flux, significa que seu código pode ser modificado e mantido mais facilmente do que se você inventasse um paradigma completamente novo.

Ter grandes ecossistemas e usar ferramentas populares e robustas bem suportadas é uma estratégia comprovada ano após ano para render resultados positivos para a empresa e a carreira do desenvolvedor e ter bibliotecas comuns ou de acordo com as normas do setor significa que você pode encontrar membros de equipes para ajudá-lo com o desenvolvimento e teste. As metodologias de desenvolvimento modernas praticamente demandam o uso de estruturas, bibliotecas reutilizáveis e componentes e APIs bem projetados.

1. Andrew S. Tanenbaum, **Computer Networks** , 2nd ed., p. 254
2. https://www.webcomponents.org/specs

Popularidade de estruturas e bibliotecas modernas

Stack Overflow, um site Web para desenvolvedores incrivelmente popular usado para perguntas e respostas (nº 57 de acordo com Alexa a partir de janeiro de 2019) rastreia uma grande quantidade de dados sobre a popularidade de diversas tecnologias e se tornou uma fonte obrigatória para desenvolvedores. Sua pesquisa mais recente continuou demonstrando a incrível popularidade do JavasScript e das estruturas e bibliotecas JavaScript:

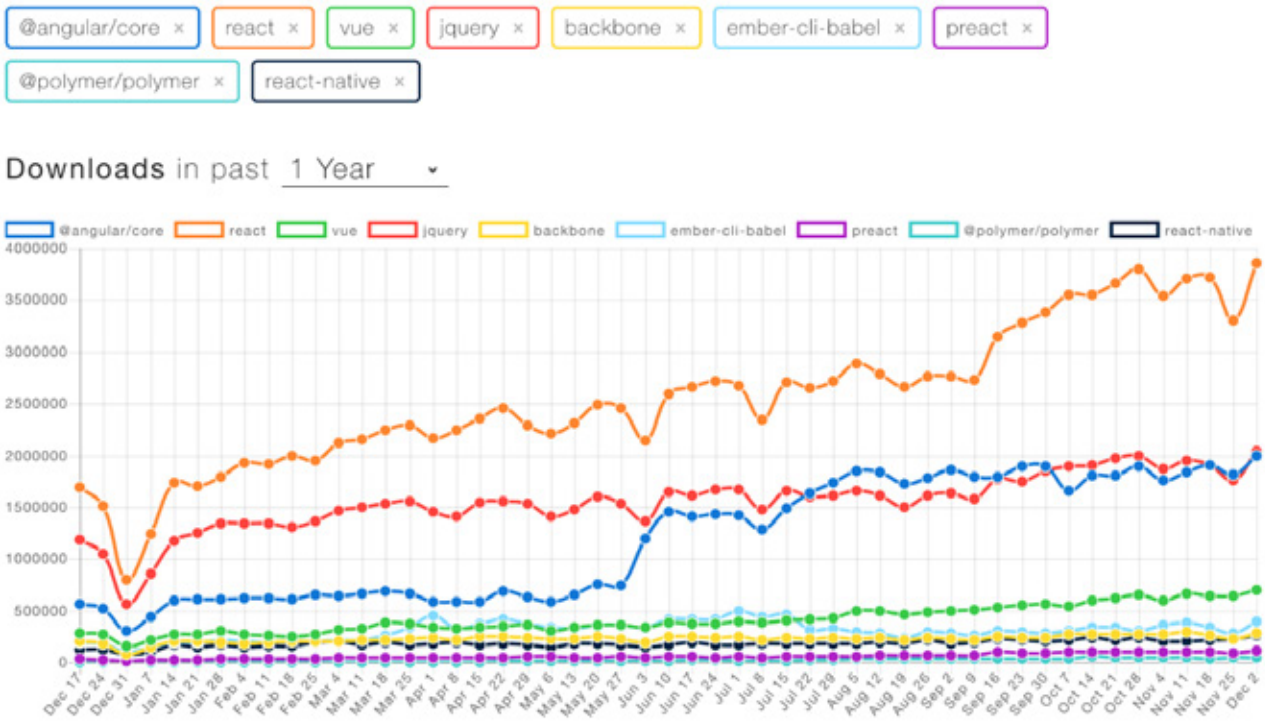


Figura 1: NPM de downloads de bibliotecas de front-end populares

De acordo com o Stack Overflow, com base no tipo de marcas atribuídas a perguntas, os oito principais tópicos mais discutidos no site são JavaScript, Java, C#, PHP, Android, Python, jQuery e HTML — não C, C++, ou linguagens mais exóticas como Ocaml ou Haskell. Se você estiver criando sites Web, muito provavelmente desejará usar tecnologias que sejam populares porque a quantidade de código aberto e produtos suportados/comerciais oferece a você a capacidade de codificar e testar mais rapidamente, o que resulta em um tempo de comercialização mais rápido.

Para os desenvolvedores, isso significa que o mundo JavaScript continua a liderar todos os outros na quantidade de desenvolvedores, e, embora tecnologias antigas como jQuery ainda sejam populares, claramente, o React e Angular são importantes e continuam crescendo. O recente, Vue, também está se tornando cada vez mais popular.

Escolhendo Angular, React ou Vue

Angular versus React versus Vue — tantas ferramentas de código aberto. Adicione a isso bibliotecas como Backbone.js e centenas de outras. Como os desenvolvedores podem atualizar seu conhecimento sobre tantas assim? Qual eles devem escolher? Até algum ponto, essa decisão se limita a escolher os editores de texto: é uma escolha pessoal e é defendida intensamente e, no fim, cada um deles pode funcionar para você.

Se sua principal preocupação é a popularidade para que não fique preso com o aprendizado de um ambiente de programação rico e complicado apenas para ver o suporte definhar, então o React é claramente a escolha ideal como a linha de tendência de longo prazo demonstra. Mas a popularidade é apenas um atributo de uma longa lista de fatores de decisão importantes.

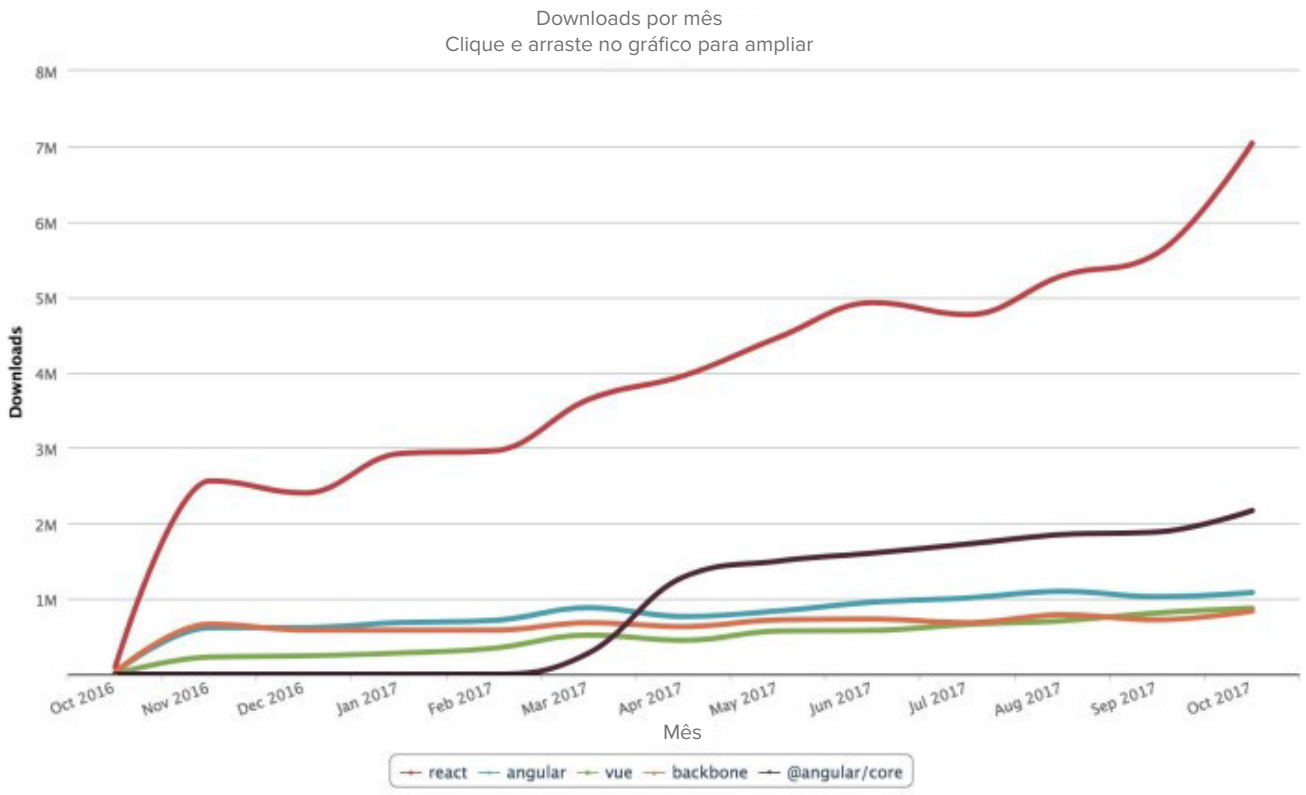


Figura 2: Linhas de tendência de longo prazo de diversas estruturas populares e bibliotecas

Vamos examinar cada um deles em um formato fácil de assimilar.

React

Melhor para: Sites Web que possuem muitos dados dinâmicos e desejam um desempenho muito alto.

O React é a biblioteca JavaScript mais popular. Desenvolvido e disponibilizado pelo Facebook em 2013, ele se tornou rapidamente importante para o desenvolvimento de grandes aplicativos Web que envolvem o manuseio dinâmico de dados. O maior ponto positivo do React ocorre quando os desenvolvedores precisam criar aplicativos complexos e desejam modularizá-los de forma que essas peças possam ser reutilizadas. O React, por sua natureza, possui vantagens de desempenho e permite que os desenvolvedores criem em JSX. O React é especialmente adequado para o gerenciamento de dados em tempo real. O Instagram e o WhatsApp, dois aplicativos Web extremamente populares, usam o React.

PRÓS	CONTRAS
Fácil de aprender. O React possui uma sintaxe relativamente simples e muito de HTML. Em contrapartida, o Angular exige uma grande curva de aprendizado e treinamento de TypeScript (definição prévia do tipo de variável). O React trabalha com HTML, CSS3 e outras ferramentas familiares fáceis de aprender.	**O React continua evoluindo.** Os desenvolvedores precisam continuar aprendendo e estar sempre atentos às mudanças do React.
O React é uma biblioteca JavaScript, não uma estrutura. Isso significa que ele oferece um método declarativo de definição de componentes de UI. Ele também pode ser facilmente integrado com outras bibliotecas.	**JSX pode ser difícil.** O React faz uso ativamente do JSX, permitindo uma combinação de HTML com JavaScript. Embora o JSX possa proteger o código contra injeções de hacking, ele envolve certa complexidade e uma curva de aprendizado.
Camadas de apresentação e dados separadas. O React oferece a separação de camadas de dados e apresentação e suporta padrões de design MVC, MVVM e Flux.	**O React não é uma estrutura completa.** A integração dessa biblioteca em uma cadeia de ferramentas do desenvolvedor (por exemplo, uma estrutura de suporte de MVC) precisa ser realizada pelo desenvolvedor.
Ligação com DOM. Os desenvolvedores não precisam ligar elementos DOM à funcionalidade. O React gerencia isso dividindo a ligação entre diversas áreas do código. O React também inclui o conceito de um DOM virtual e apenas os itens que mudaram são atualizados e exibidos para o usuário.	**Documentação.** Uma das reclamações mais frequentes sobre o React é a falta de documentação oficial.

Componentes reutilizáveis. O React oferece a capacidade de reutilizar componentes de código de qualquer nível a qualquer momento. Isso economiza muito tempo durante o desenvolvimento e facilita para desenvolvedores gerenciarem atualizações. Qualquer mudança feita em um componente não tem	
Fluxo de dados unidirecional. O React utiliza a ligação de dados descendente. Isso ajuda a garantir que todas as alterações feitas na estrutura filha não acabe afetando seus pais. Isso torna o código mais estável e tudo que o desenvolvedor precisa fazer para alterar um objeto é modificar seu estado e aplicar atualizações. O React aplica o fluxo de dados de cima para baixo.	
Interoperação com outros componentes. O React pode ser usado como o componente de Exibição do Angular ou outra estrutura, porque ele não o vincula a uma pilha de tecnologia específica.	
Compatível com mecanismo de busca. O React pode ser executado no servidor e o DOM virtual será retornado e renderizado para o navegador como uma página da Web normal.	
Suporte à programação funcional. O React pode usar	

Angular

Melhor para: Desenvolvedores que não desejam brigar com a integração de bibliotecas em outras

O Angular, ou, anteriormente, AngularJS, é mais antigo do que o React e é o segundo mais popular em nossa análise. Ele foi criado e é mantido pela Google e foi desenvolvido para atender à necessidade de uma **estrutura completa** que pudesse lidar com aplicativos de uma única página. Um aplicativo de uma única página (SPA) é um aplicativo da Web ou site da Web que atualiza informações e dados reescrevendo dinamicamente a página atual em vez de carregar novas páginas inteiras a partir de um servidor. Portanto, o aplicativo se parece mais com um aplicativo de PC desktop, mas também funciona bem em dispositivos móveis se projetado adequadamente. Todos os códigos são carregados com o carregamento de uma única página, ou apenas páginas parciais são carregadas dinamicamente quando a página é atualizada. O Angular pode ter uma curva de aprendizado relativamente longa porque é realmente uma estrutura de desenvolvimento completa, mas o Angular em si não contém componentes de UI. A Sencha preenche essa lacuna com elementos de UI pré-testados e desenvolvidos consistentemente com seu produto ExtAngular.

PRÓS	CONTRAS
Aplicativos de página única (SPAs). O Angular foi projetado exatamente para esse cenário.	**Sintaxe complexa.** O Angular pode ser complicado e não se encaixa no estilo de criação de código de todos os programadores.
Aplicativos Web progressivos (PWAs). Se deseja criar aplicativos que se pareçam com aplicativos nativos em dispositivos móveis, tablets ou desktops, o Angular possui isso.	**Curva de aprendizado mais elevada.** O Angular pode ser difícil de aprender, embora a documentação e a ajuda disponíveis ajudem a mitigar isso até certo ponto.
Estrutura de escala total. O desenvolvedor não precisa integrar o Angular em suas próprias ferramentas e estrutura de MVC. Também suporta o padrão de design Flux.	**Compatibilidade retroativa difícil.** Mover aplicativos do Angular para versões mais recentes pode levar um tempo, uma vez que a portabilidade pode ser difícil devido a mudanças de versão.
Documentação. Diferentemente de muitas tecnologias Web, o Angular possui uma documentação extensa com muitos vídeos no YouTube disponíveis. Isso ajuda a encurtar a curva de aprendizado	**Alta evasão de JavaScript.** Como a Google e o mundo do código aberto está constantemente melhorando o Angular, o desenvolvedor precisa manter-se atualizado sobre novas versões ou arriscar desenvolver aplicativos que exigirão uma portabilidade significativa no futuro se precisarem ser alterados.
API TransferState universal e DOM. Com esse novo design, o código pode ser compartilhado entre o servidor e o cliente, o que pode melhorar o desempenho.	
Otimizador de criação. Como um bom compilador de otimização, ele remove todos os códigos de tempo de execução desnecessários, reduzindo o tamanho do aplicativo e melhorando o desempenho.	
Ganchos de roteador. Agora os ciclos de roteador podem ser rastreados desde o início das proteções de execução até a ativação ter sido concluída.	
Ligação de dados de duas vias e MVVM (Modelo-Exibição-Exibição-Modelo). A ligação bidirecional minimiza o risco de erros e permite um comportamento singular do aplicativo. Para padrões de design MVVM, ele tem a capacidade de permitir que os desenvolvedores trabalhem separadamente na mesma seção de aplicativo com o mesmo conjunto de dados.	

A Google oferece suporte ao Angular. Assim como o Facebook usa o React, a Google usa e continua desenvolvendo o Angular. Ele também é usado pelo Google Adwords, portanto, o suporte de longo prazo é provável.	

Vue

Melhor para: Desenvolvedores com aplicativos mais simples e curva de aprendizado mais baixa.

O Vue é recente e está ganhando popularidade em um ritmo extraordinário. Entre novembro de 2016 e outubro de 2017, uma estimativa (npm de downloads) demonstra que o Vue está crescendo em uma taxa incrivelmente alta de 13.933,4%, mas os números de download absolutos demonstram o React com mais de 7 milhões de downloads em outubro de 2017 em comparação com aproximadamente 900.000 do Vue, com o Angular nesse intervalo com mais de 2 milhões de downloads de npm.
De maneira interessante, todas as estruturas e bibliotecas JavaScript continuam crescendo em ritmos acelerados. A origem desses dados é npm-stat: https://npmcharts.com/

O Vue fica na parte intermediária entre uma estrutura completa e "apenas uma biblioteca". Funções não essenciais como roteamento, gerenciamento de estado, cadeias de ferramentas de criação e o CLI são externas, mas todas são oficialmente mantidas, bem documentadas e projetadas para trabalharem em conjunto. No entanto, você não precisa usá-las todas. O Angular aplica uma determinada estrutura sobre como o código é organizado; o Vue não.

De acordo com o notável desenvolvedor John Hannah:

"O React e o Vue são bem semelhantes, embora haja algumas diferenças.... Isso faz sentido, uma vez que Evan You, o desenvolvedor do Vue, usou o React como uma de suas inspirações. Eles são muito mais parecidos entre si do que são com o Angular, por exemplo." Na documentação do Vue, vemos que tanto o React quanto o Vue:

- Utilizam um DOM virtual por motivos de desempenho
- Oferecem componentes de exibição reativos e moduláveis
- Mantêm o foco na biblioteca central, com considerações como roteamento e gerenciamento de estado global gerenciadas por bibliotecas complementares.

Fonte: John Hannah, https://jsreport.io/how-is-react-different-from-vue/

Como o React, o Vue enfatiza o desempenho, mas, por ser menor, é mais fácil de aprender inicialmente.

PRÓS	CONTRAS
Fácil de aprender. Usa modelos em vez de JSX. Esses modelos são extensões do HTML, não JavaScript.	**Ecossistema muito menor.** Tanto o React quanto o Angular, por serem mais maduros, têm ecossistemas muito maiores do que o Vue.
O Vue, como o React, é uma biblioteca JavaScript, não uma estrutura. Isso significa que ele oferece um método declarativo de definição de componentes de UI. Ele também pode ser facilmente integrado com outras bibliotecas.	**Menos evasão de JavaScript.** O Vue é uma biblioteca e os desenvolvedores buscam manter todas as bibliotecas complementares atualizadas.
Personalização mais simples. A personalização é feita por meio de marcas ou CSS, com o padrão sendo de marcas de estilo mais simples.	
Gerenciamento de estado, roteamento etc. Isso é realizado por bibliotecas externas (mas mantidas de maneira coerente). Não aplica um design/organização de código rígidos.	
Suporte a diversos padrões de design. Isso pode ser realmente definido como "não impõe um padrão de design sobre você," mas o Vue não oferece suporte a MVC ou Flux.	

Desenvolvedor focado em tabela de exibição rápida:

A programação não é apenas uma profissão desafiadora, é uma busca baseada em linguagem intensamente pessoal que exige pessoas inteligentes, quase sempre sob enormes pressões de cronogramas e recursos. A seleção de quais ferramentas JavaScript usar é, como dizemos, intensamente pessoal e geralmente reflete como os desenvolvedores pensam sobre os problemas. Resumidamente, é um negócio voltado para as pessoas e normalmente um "esporte em equipe" (portanto, a necessidade de ferramentas como Assembla para gerenciar o código-fonte em diversos projetos e muitas pessoas).

Essa tabela de exibição rápida pode ser mais útil para você do que uma análise técnica, porque ela divide essas três tecnologias em como elas afetam a capacidade de uma pessoa de realizar o trabalho e oferece uma classificação lado a lado. Essas classificações são uma questão de opinião e podem mudar com base na familiaridade de um desenvolvedor, estilo de codificação, tipo de padrões de design que eles usam, na disponibilidade das outras pessoas ao redor deles que podem fornecer ajuda e orientação e em provavelmente uma centena de outros fatores.

ATRIBUTO	REACT	ANGULAR	VUE
Popularidade	1	2	3
Apoiadores	Facebook	Google	Laravel e Alibaba
Biblioteca (L) ou Estrutura (F)?	L	F	L
Estilo de codificação voltada para objeto	✔	✔	✔
Estilo de codificação de processo imperativo/funcional	✔	✔	✔
Se precisar de orientação, estrutura e uma ajuda		✔	
Curva de aprendizado	Moderada	Alta	Baixa
Velocidade de codificação (a velocidade com que você implanta)	Média	Mais Lenta	Mais Rápida
Precisa de flexibilidade	✔		
Grande ecossistema, muitos pacotes	1º	2º	3º
Ama JavaScript, apenas quer o JavaScript	✔		
Criação de grandes aplicativos	✔	✔	
Disponibilidade de desenvolvedores	✔	✔	
Aplicativos de página única		✔	
Desempenho mais alto	1º	3º	2º
Qualidade e quantidade de documentação	2º	1º	3º

Suporta código JS legado?	Não, precisa reescrever em JSX	Sim, mas precisa de conversão para TypeScript	Sim, mas pode precisar de conversão para a sintaxe Vue.js

Fonte: Davison Pro, https://medium.com/@davisonpro/react-js-vs-angular-7a7bed92b5f6 e outros

Criando nos ombros de gigantes: Por que reinventar a roda o atrasa

Desenvolver estruturas sofisticadas, bibliotecas reutilizáveis e especialmente widgets de componentes de UI exige um tempo valioso e tempo é dinheiro. A luta para encontrar e combinar diferentes ferramentas gratuitas e de código aberto (FOSS) pode ser cara e demorada e pode resultar em decepção, uma vez que algumas ferramentas podem ser incompatíveis ou exigir camadas de ajustes adicionais.
Hoje, os principais indicadores de desempenho mudaram. O barulho incansável para CIOs e chefes de desenvolvimento não é das linhas de código, do número de pontos de função entregues e de outras métricas de engenharia. Eles deram espaço para objetivos comerciais e, como é assim que os CIOs e cada vez mais CTOs estão sendo classificados, os desenvolvedores também estão sendo classificados por seu impacto nos negócios. Esses objetivos comerciais agora incluem o lucro e reduções de custo com maior eficiência; normalmente eles incluem o tempo de comercialização (TTM, ou tempo para os DevOps implantarem uma solução funcional), menos barreiras para entrada (LBE, começando com ferramentas abertas e código aberto ou usando downloads de avaliação gratuita para protótipo), e custo total de propriedade (TCO; quanto essa tecnologia me custará em curto e longo prazo).

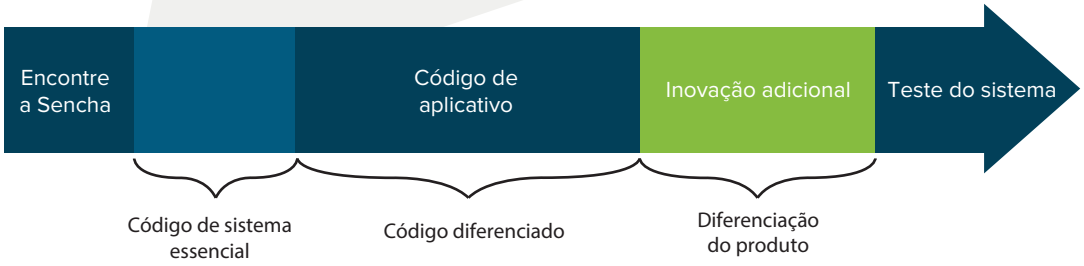
Como resultado, os desenvolvedores estão se tornando mais estratégicos no que importa para eles tornarem-se conscientes e alinhados com as estratégias de negócios de seus empregadores. Mas, mais do que isso, as exigências aumentaram: agora, dispositivos móveis, telefones, tablets, PCs desktop e até mesmo decodificadores e smartTVs são alvos potenciais para implantação.

Felizmente, existem soluções: adotar uma boa estrutura ou conjunto de bibliotecas pode economizar o tempo, dinheiro, esforço e dores de cabeça dos desenvolvedores e fornecer futuros brilhantes. Por exemplo, adotar o React, Angular, ou Vue como a base dos esforços de desenvolvimento de uma pessoa oferece diversos benefícios e esses benefícios podem ser selecionados com base em requisitos e circunstâncias.
Usar widgets de UI robustos de plataforma cruzada como Sencha Ext JS, ExtReact, ou ExtAngular pode economizar muitos messes de busca do que você precisa usar e tempo de codificação e teste. A Sencha garante que todos esses elementos funcionem com a grande maioria de navegadores Web, e fazer seu aplicativo Web funcionar em plataformas Android e iOS sem mais esforço pode evitar que você tenha que aprender linguagens difíceis e complicadas. Apenas use o JavaScript.

Para preservar a seleção do desenvolvedor, é bem possível combinar e corresponder componentes de UI de código aberto e produtos comerciais da Sencha. No entanto, uma maior produtividade é resultado da busca de tudo que você precisa de um fornecedor, o que reduz o tempo de pesquisa do desenvolvedor.

Desenvolvimento usando apenas ferramentas de código aberto gratuitas

Desenvolvimento usando produtos Sencha



Visite www.sencha.com para saber mais, acesse downloads de avaliação gratuita de 30 dias ou uma licença comercial limitada Community Edition.

Resumo

COMECE O DESENVOLVIMENTO DO APLICATIVO ANTECIPADAMENTE	AJUSTADO PARA A EFICIÊNCIA DO DESENVOLVIMENTO	REALIZE OS NEGÓCIOS EM UMA ETAPA
Concentre o desenvolvimento do software nas camadas do aplicativo, liberando recursos para realizar a diferenciação do produto.	O Software e a documentação são disponibilizados imediatamente, pré-testados, pré-qualificados, pré-integrados e ajustados para melhor eficiência de desempenho do código.	A Sencha mantém todos os seus produtos, fornecendo uma fonte para todo o Suporte técnico– uma linha de ajuda para quando seu aplicativo estiver em risco!