



Instituto de Tecnologia

Guia para começar do zero uma carreira em

Desenvolvimento de Software em 2022

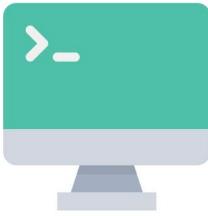




Sumário

- **Introdução**
- **Pra quem é este guia?**
- **Existe uma forma de se pensar e agir que é comum para quem desenvolve software?**
- **As possíveis áreas de atuação**
- **Será que sou realmente capaz?**
- **Como e quando começar meus estudos**
- **Quando posso me considerar pronto para conquistar uma 1ª oportunidade?**
- **E agora? O que vocês sugerem para o meu próximo passo?**
- **Comece agora com o curso gratuito no ITuring!**





Introdução

Enquanto estimativas apontam um **déficit de mais de 400 mil profissionais de tecnologia¹** no Brasil e **40 milhões de vagas não preenchidas ao redor do mundo²**, os salários médios podem ultrapassar os **R\$ 11.000,00 mensais** para desenvolvedores de software com alguma experiência no Brasil, e até 20 a 60 mil reais mensais caso trabalhe remotamente para uma empresa no exterior. São muitas vagas abertas para poucos profissionais qualificados, e isso faz com que as empresas ofereçam salários altos na esperança de atrair a maior quantidade possível de talento na área. E esses números só tendem a aumentar, visto que cada vez mais a sociedade depende da tecnologia para tarefas básicas, além de empresas e setores da indústria e de serviços que estão se modernizando cada dia mais.

Mas, com tantas vagas abertas e salários tão altos, por que mais pessoas não estão buscando uma carreira na área?

Um dos motivos é a percepção disseminada na sociedade de que programação e tecnologia no geral é uma área “difícil”, algo que somente pessoas “inteligentes” conseguem aprender.

Neste guia, nossa meta é desmistificar estas ideias e mostrar que, com um bom plano de estudos, ajustes comportamentais e de mentalidade e muita dedicação e perseverança, qualquer pessoa pode aprender a programar e conquistar uma primeira oportunidade na área. Sim, **qualquer** pessoa! Independente se a pessoa fez algum curso superior ou não. Selecioneamos de forma estratégica conteúdos sobre habilidades técnicas e não-técnicas necessárias, quais as possíveis áreas de atuação, melhores práticas na busca por vagas e construção de portfólio, entre outras coisas, com o objetivo de gerar uma transformação real em sua carreira.

Ao pensar em buscar uma nova carreira, em especial a de desenvolvimento de software, sempre surgem muitas dúvidas, angústias e inseguranças. Pensamentos como “será que sou capaz?”, “isso é realmente pra mim?” ou “não sou tão bom quanto os outros” nos paralisam e nos impedem de dar o passo decisivo que faz uma pessoa evoluir de estudante para desenvolvedor de software profissional. Mas não se sinta sozinho, estes pensamentos intrusivos são muito comuns e praticamente todo programador profissional passou por situações similares em seus primeiros anos de carreira. Iremos te dar todo o apoio necessário para sanar quaisquer dúvidas de que você seja capaz de aprender programação e garantir que você tenha tudo o que é necessário para conquistar uma primeira oportunidade na área.

No mais, com muito pé-no-chão e astúcia, desejamos uma boa leitura e estaremos ansiosos para tê-lo como colega de profissão em breve!

1. <https://canaltech.com.br/carreira/pesquisa-preve-carencia-de-408-mil-profissionais-de-ti-ate-2022-189998/>
2. <https://www.daxx.com/blog/development-trends/software-developer-shortage-us>

Pra quem é este guia?



Preciso ser um "gênio" para aprender a programar?

Resposta curta: NÃO!

Esta é uma das dúvidas mais comuns de quem vislumbra começar uma carreira em programação. E o mais engraçado é o seguinte:

- Na visão de **alguém de fora** da área de tecnologia, o estereótipo de quem desenvolve software é aquela pessoa reclusa, inteligente ao extremo, que não se comunica muito com os demais colegas de empresa e fica até de madrugada digitando furiosamente códigos aparentemente incompreensíveis. E bem, talvez esse fosse o perfil dos primeiros cientistas da computação, porém a realidade hoje é totalmente diferente.
- Na visão de **quem está dentro** da área de tecnologia, a visão é simplesmente o oposto. Quem desenvolve software trabalha em equipe, em torno de um objetivo em comum. Não é preciso ser gênio, mas é necessário esforço e dedicação. É um processo diário em que a pessoa vai se desenvolvendo, colaborando, aprendendo e expandindo suas habilidades.

Ao invés de um trabalho solitário, hoje, equipes de desenvolvimento de software em todo o mundo celebram uma comunicação clara, transparente e constante entre o time. Ao invés de horas e horas digitando sem parar, na verdade, programadores passam horas e horas lendo documentações, tutoriais ou colaborando com o time. Ao invés de saber "de cabeça" algoritmos complexos e estruturas de dados mirabolantes, um desenvolvedor precisa pensar em como os usuários irão interagir com aquele produto, se seus colegas de equipe irão compreender seu código, e como aquela tarefa afeta o negócio como um todo.

Com décadas de aprimoramento nas ferramentas e técnicas na área de tecnologia, além da ampla gama de informação que a internet proporcionou para a maioria das pessoas, o foco da programação migrou de dominar temas extremamente complexos como algoritmos e estruturas de dados para construir produtos que são amigáveis aos usuários, fáceis de dar manutenção e principalmente, fáceis de expandir e escalar.

Então algoritmos e estruturas de dados não são importantes? Sim, eles são importantes. Porém, mais importante que isso é o processo constante de aprendizado. Lembre-se deste outro, programar é aprender constantemente. E quanto mais você aprender, mais coisas complexas você vai dominar.

O que antes pra você era complexo, hoje não é mais. O que antes era um algoritmo avançado que para você era impossível de desenvolver, hoje não é mais. É este o processo que todo programador aplica. Diferente de qualquer outra profissão, programação é uma das áreas em que você mais aprende, o tempo inteiro. Em um momento você vai precisar de um algoritmo mais complexo para resolver um determinado problema. A-ha! Você já sabe, você pegou o ritmo de aprendizado constante. Neste momento, você vai parar o que está fazendo e vai estudar aquilo, aprender e aplicar. É este o processo. E é por isso que não existe programador gênio, mas apenas programadores que aprendem constantemente, se esforçam, se dedicam e colaboram.

Além do processo de aprendizado constante, hoje, um desenvolvedor que se preocupa se seu código vai ser compreendido por alguém que começou a trabalhar na empresa há 5 dias é **muito mais valioso** do que um que consegue escrever um algoritmo de "Quicksort" em uma única linha sem consultar o Google.

A área de tecnologia é predominantemente dominada por pessoas jovens. Será que já passei da idade?

O sentimento de "não tenho tempo de aprender tudo o que o mercado pede" é a coisa mais comum que existe em quem está tentando entrar na área, principalmente entre pessoas a partir dos 25 anos. Portanto, não fique muito preocupado com isso, pois praticamente todos passam por esse sentimento. Esta sensação é comum e muitas vezes vem do excesso de informação e da falta de um plano de estudos bem estruturado. Se você pesquisar "como me tornar programador em 2022" na internet, receberá uma enxurrada de conselhos e dicas, em sua maioria incompatíveis entre si.





Ainda nessa questão da sensação de sobrecarga, de ter muita coisa pra estudar e talvez pouco tempo pra aprender, é muito importante ter foco e estudar de forma estratégica, mantendo a calma e construindo um projeto por vez, do começo ao fim. Cada projeto desenvolvido servirá principalmente para duas coisas: provar para si mesmo que você é capaz de construir algo real e completo e servir de portfólio para potenciais recrutadores. É essencial que você aprenda os conceitos básicos inicialmente, e depois vá lentamente experimentando cada uma das áreas mais especializadas, sempre observando os temas que têm mais afinidade e se aprofundando neles. Assim, futuramente, você terá como objetivo conseguir uma primeira vaga especificamente nas suas áreas favoritas, o que aumentará consideravelmente sua chance de sucesso. Ao longo deste e-book iremos detalhar quais as melhores formas de montar seu plano de estudos, e quando você deve se considerar “pronto” para tentar uma primeira oportunidade.

Um fato que percebemos na área e também nas empresas é que não existe idade para começar, pois o que depende mesmo é do seu comportamento e mentalidade. Existem pessoas que começaram a programar muito cedo, mas que se estagnaram e nos seus 40 anos ainda atuam como desenvolvedor pleno. Em contrapartida, há também casos de pessoas que começaram com 30 anos de idade e em 4 anos já são gerentes de engenharia.

Outro ponto importante para se considerar no caso de pessoas mais velhas, é que em sua maioria estão buscando uma migração de carreira. Isso significa que existe uma experiência prévia e habilidades já desenvolvidas que podem ser muito úteis na área de tecnologia, e muito atrativas para as empresas.

“Soft skills” (habilidades sociais/interpessoais) como comunicação, gestão de tempo, inteligência emocional e liderança estão sendo cada vez mais solicitadas pelas empresas, e se você possui experiência nessas competências por conta de sua carreira atual, não só será ainda mais atrativo para o mercado considerando uma vaga de entrada, como também progredirá em sua nova carreira mais rapidamente. Existem diversos casos de profissionais mais velhos que após se completarem cursos de programação, não só conseguiram entrar na área de tecnologia como também conquistaram oportunidades melhores muito mais rápido por serem mais maduros, se comunicarem melhor, e no geral possuírem as competências interpessoais mais desenvolvidas.



O ponto de não existir idade é justamente por conta de que a carga de aprendizado e expertise que você pode acumular muitas vezes independe do seu tempo de carreira. Quem tem 30 anos e está começando do zero, pode aprender muito e obter uma expertise muito maior do que quem começou desde os 20 anos. Ou seja, temos “tempo de vida” mais do que suficiente para aprender, avançar e obter sucesso na área. Seu comportamento e atitude é que definirão sua capacidade de chegar lá.

Nunca é tarde para começar, a única coisa que você precisa é a vontade de aprender e a perseverança de não desistir frente aos erros.



E os recrutadores? Não vão achar ruim de eu ter começado tão tarde?

Uma coisa extremamente importante em qualquer carreira é assumir que a maioria das pessoas no mundo são bem intencionadas. E o que isso quer dizer? Significa considerar que a maioria dos recrutadores estão bem intencionados, ou seja, eles não vão olhar sua idade como um ponto negativo. Portanto, considere que, salvo em casos particulares, um recrutador não vai considerar sua idade como fator de contratação, mas sim o seu comportamento, suas atitudes e seu nível de experiência necessário para a vaga. Foque nisso e livre-se dessa preocupação! Mas calma, nem tudo é perfeito, pois infelizmente ainda existem empresas que discriminam por idade. Neste caso, quando você se deparar com uma empresa desse tipo em seu início de carreira, tenha a certeza que essa provavelmente **não é uma boa empresa** para se trabalhar e que, com certeza, essa empresa terá problemas graves caso não atualize seus valores para condizer com a sociedade moderna.



Existe uma forma de se pensar e agir que é comum para quem desenvolve software?

Existem vários perfis diferentes de pessoas que trabalham com programação. Mas existe um padrão muito comum observado na maioria dessas pessoas. É importante saber desses traços para que sirva de norte para o seu desenvolvimento e para você saber o que você já tem e o que precisa melhorar. Veja alguns desses traços:



Mentalidade “problem solver”

O termo “problem solver” em inglês significa, literalmente, “resolvedor de problemas”, e resume muito bem o propósito de um programador dentro de um projeto. Muitas pessoas fora da área de tecnologia têm a visão de que o trabalho de um programador é escrever código, mas isso não é completamente verdade. A programação é apenas uma ferramenta, e seu trabalho será resolver os problemas dos usuários, fazendo uso das ferramentas necessárias. Portanto, a habilidade de identificar e, principalmente, propor e implementar soluções para os problemas que surgem em um projeto é a característica mais importante que um programador pode ter. Mas lembre-se: esta habilidade também pode ser praticada e melhorada com estudo, portanto, esteja preparado para resolver muitos problemas. Trabalhe em você essa vontade e paixão por resolver problemas.



Prática intensa

A única forma de aprender a programar é programando. Ler livros, assistir vídeos ou fazer resumos são todas ferramentas úteis para seu aprendizado, mas não substituem a necessidade de **realmente** abrir o editor de códigos e programar. Um programador deve programar diariamente, no trabalho ou fora dele, pois essa é a única forma de você realmente se desenvolver na área. Quase todo programador bem-sucedido possui diversos projetos pessoais, que servem para aprimorar habilidades diferentes daquelas que já são diariamente praticadas no trabalho. Portanto, prepare-se para a programação passar a ser sua atividade favorita do dia-a-dia.





Aprendizado constante

Todo programador é um eterno estudante, e nenhum programador é um mestre. Nessa área, não existe um “teto” de conhecimento onde você pode se considerar conhecedor o suficiente de um assunto para não precisar mais estudar. A tecnologia evolui em uma velocidade assombrosa, e muitos conhecimentos que eram essenciais cinco anos atrás, hoje são obsoletos, enquanto novas ferramentas que possuem sequer um ano de existência serão as novas habilidades essenciais dos próximos cinco anos. Ser um programador é ter consciência que você jamais poderá se acomodar em sua busca por conhecimento, é sentir prazer em aprender coisas novas, e jamais ir dormir sem ter aprendido algo novo.



Resiliência

A habilidade de se recuperar de erros. Ser **resiliente** é essencial para alguém nesta área, pois todo programador irá errar constantemente durante sua jornada. Lógicas equivocadas, erros de digitação, falta de entendimento do escopo, entre outras coisas fazem parte do dia-a-dia da programação, e não devem ser encaradas como um “desastre”. É essencial que, ao errar, você comprehenda seu erro e aprenda com ele, para não voltar a cometê-lo no futuro. Porém, jamais crie a ilusão que você não cometerá mais erros, pois **isso é simplesmente impossível**. Programas de computador são estruturas complexas e por conta disso estão fadadas a possuírem partes defeituosas. Isso é amplamente aceito na indústria, e a maioria das empresas possuem diversos mecanismos para encontrar, solucionar e prevenir estes defeitos, como setores inteiros de controle de qualidade, onde o software será profundamente testado antes de ser disponibilizado ao público. Qualquer programador, do mais júnior ao mais sênior, erra constantemente. A diferença é que, pessoas experientes em programação já esperam errar, e sabem agir adequadamente quando isso acontece. **Lembre-se:** errar não é a única forma de evoluir, mas provavelmente será a forma mais frequente. Portanto, **acostume-se com isso!**



Trabalho em equipe

Dificilmente pessoas que programam trabalham sozinhas. Atualmente os times de desenvolvimento de software são multidisciplinares, sendo compostos por diversos programadores, designers, profissionais da área de negócios, testadores de software, profissionais de infraestrutura, entre outros. Faz parte da função a comunicação constante e a colaboração efetiva, pois softwares de qualidade não são construídos apenas por bons programadores, mas muito mais por bons times. Quem quer trabalhar com programação deve considerar imprescindível a empatia em relação aos colegas, no sentido de estar sempre preocupado se os outros programadores conseguirão compreender o código que foi escrito, agora ou no futuro, se aquele código é fácil de modificar e estender, e se aquele código vai continuar funcionando pelos anos que se passam. Existem dois conceitos não-técnicos muito importantes sobre trabalho em equipe para programadores: o conceito da **dívida técnica**¹, que é o custo “oculto” de escolher soluções mais fáceis agora, porém que irão gerar retrabalho no futuro, e o **fator ônibus**², que é o risco envolvido na perda de uma pessoa chave da equipe. Para evitar essas situações, é importante que a pessoa que está programando escreva cada linha de código pensando não somente em solucionar o problema, mas também em facilitar o trabalho de seus colegas de equipe, evitando ao máximos “atalhos” que gerem dívida técnica e não tendo uma atitude centralizadora onde grande parte do conhecimento chave para o projeto fica concentrado em uma única pessoa.

1. <https://www.infoq.com.br/news/2009/10/dissecting-technical-debt/>

2. https://pt.wikipedia.org/wiki/Fator_%C3%B4nibus

As possíveis áreas de atuação

Com o passar dos anos, a área de tecnologia foi evoluindo e crescendo de complexidade. Hoje, a frase “trabalho com programação” por si só não diz muita coisa, pois essa pessoa pode trabalhar tanto construindo aplicativos e sites quanto desenvolvendo sistemas de controle para carros que dirigem sem a necessidade de motoristas.

Para quem está começando, é **muito importante** ter uma noção inicial das possíveis áreas de atuação para saber qual você se identifica mais e a partir daí traçar um plano de estudos em torno de qual área pretende seguir. Porém, ainda assim, você nunca saberá se gosta de algo ou não até que tente fazê-lo. Portanto, é importante praticar um pouco de cada área no início de seus estudos, para que você descubra com quais tecnologias você possui mais afinidade e possa decidir de forma mais certeira quais vagas disputar. Vamos te ajudar nisso e te mostrar como você pode encontrar as melhores oportunidades.



Os tipos de trabalhos e realidade salarial

Programador, desenvolvedor, engenheiro de software... todos esses termos, na prática, significam a mesma coisa: um profissional cujo objetivo é resolver problemas através de programas de computador. Vamos utilizar o termo desenvolvedor como developer, em inglês.

Basicamente, developers podem trabalhar em 3 principais modalidades:

Freelancers



Você presta serviço para uma empresa, mas não é contratado diretamente por ela. Ou seja, você não é considerado um membro do time da empresa, mas uma pessoa de fora que colabora em alguma frente por um tempo determinado. Normalmente todo freelancer tem um PJ, ou seja, uma própria empresa que te representa ao prestar serviços para uma outra empresa.

Empregado

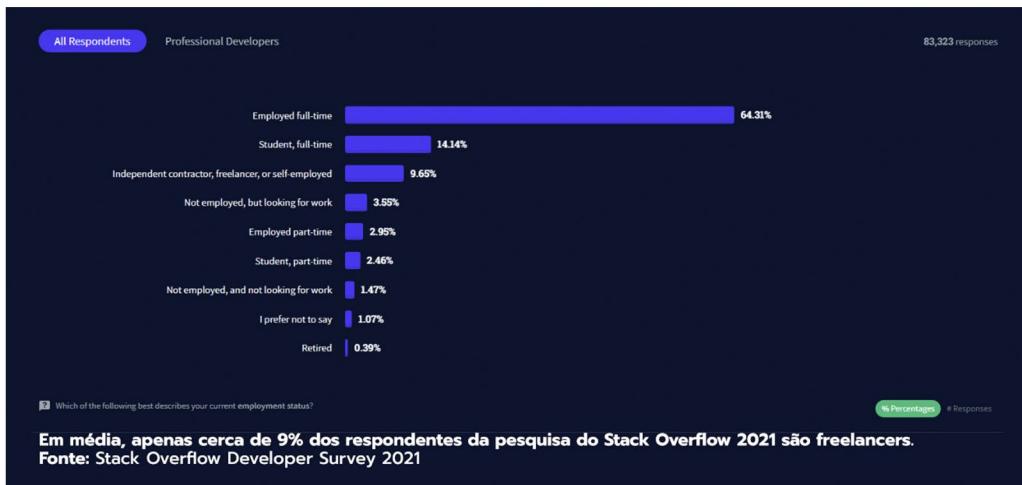


Aqui, diferente do freelancer, você é considerado um membro do time, parte integral da empresa, seja ela pública ou privada. Geralmente o tipo de contrato é CLT, mas existem empresas privadas que contratam a pessoa como PJ. Quando um empregado é contratado como PJ, normalmente eles definem no contrato um período fixo de 1 ano, mas sem data de expiração, ou seja, ele se renova a cada ano caso nenhuma das partes se manifeste em sentido contrário. O que diferencia o empregado do freelancer é que o freelancer é visto como um agente externo, enquanto que um empregado é parte integral do time da empresa e participa de reuniões gerais e

Empreendedor



E por último, por que não começar seu próprio negócio? Muitas grandes empresas ou softwares começaram como um projeto passional de uma ou algumas poucas pessoas que programavam. Você pode se juntar a um amigo e desenvolver, testar e validar uma ideia com o seu público alvo, ou seja, o seu mercado consumidor. Mas não se limite a um produto para o consumidor final. Seu empreendimento também pode ser uma empresa de consultoria que desenvolve software para outras empresas. Enfim, no empreendedorismo são muitas possibilidades!



Mas qual a melhor modalidade para quem está começando?

Isso depende muito da pessoa e com o que ela se identifica mais. Em termos salariais, freelancers e empreendedores são os que mais sofrem variação. Como freelancer, você pode faturar de mil reais mensais até dez ou quinze mil, dependendo de sua experiência, quantidade de clientes, qualidade de trabalho ou época do ano. Como empreendedor, provavelmente você não terá retorno algum nos primeiros meses ou até anos de sua empresa, mas, caso a empresa cresça e consiga mais clientes, os ganhos costumam ser muito maiores que os de qualquer funcionário. Empregados têm a remuneração mais previsível, e isso pode facilitar o planejamento para pessoas que preferem ter alguma previsibilidade e segurança. Também é válido lembrar que nada te impede de testar várias modalidades de trabalho ao mesmo tempo: você pode ser empregado em uma empresa durante o dia e desenvolver projetos como freelancer após o trabalho, ou trabalhar como freelancer enquanto tenta fazer sua empresa crescer. A área de tecnologia é atrativa justamente por haver tantas possibilidades diferentes de trabalhar, criar e ganhar dinheiro, portanto, aproveite!

• Realidade Salarial

Se você se interessou por programação, provavelmente os altos salários são um dos motivos, junto com a grande oferta de vagas. Separamos aqui algumas faixas salariais para diferentes cargos e níveis dentro da área de desenvolvimento de software, na região de São Paulo. Os dados são do Glassdoor, em janeiro/2022:

Desenvolvedor Fullstack

Júnior: R\$ 4.095/mês

Pleno: R\$ 5.724/mês

Sênior: R\$ 9.037/mês

Desenvolvedor Frontend

Júnior: R\$ 4.055/mês

Pleno: R\$ 7.186/mês

Sênior: R\$ 11.544/mês

Desenvolvedor Backend

Júnior: R\$ 3.230/mês

Pleno: R\$ 7.688/mês

Sênior: R\$ 11.580/mês

Desenvolvedor Mobile

Júnior: R\$ 4.641/mês

Pleno: R\$ 6.729/mês

Sênior: R\$ 11.244/mês

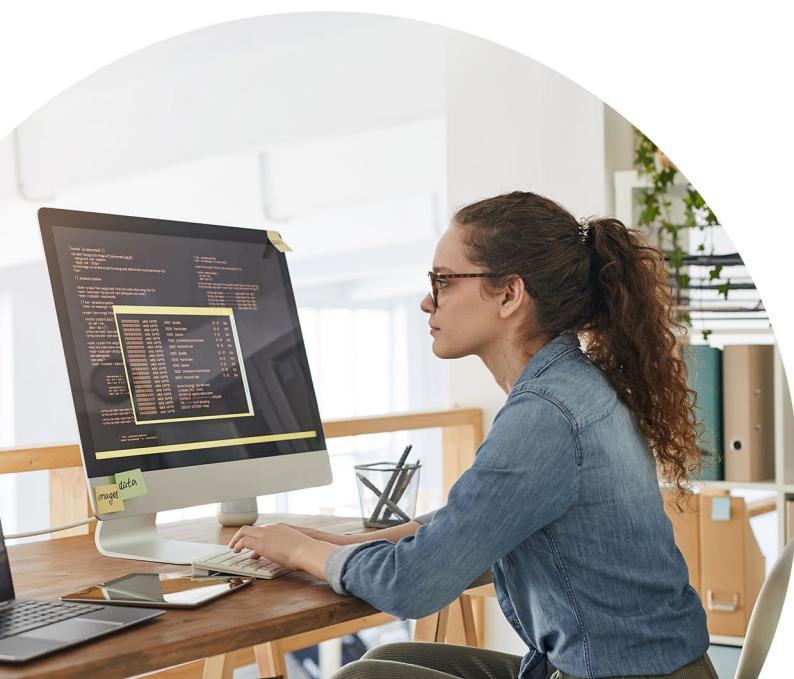
Desenvolvedor DevOps

Júnior: R\$ 3.800/mês

Pleno: R\$ 7.909/mês

Sênior: R\$ 9.624/mês

A pandemia e o trabalho remoto no Brasil e no mundo



Em 2020, por conta da pandemia de COVID-19, que afetou as empresas do mundo todo, observamos um movimento que já vinha acontecendo nos últimos anos ser acelerado: o do trabalho remoto.

A área de tecnologia foi uma das primeiras a se aproveitar das novas tecnologias de conferências por vídeo, chats em tempo real, ferramentas de trabalho colaborativo e no geral, o aumento exponencial da velocidade média de internet na casa da maioria dos trabalhadores para experimentar modelos de trabalho remoto: cada profissional trabalha em sua própria casa, reuniões são feitas através de programas como Zoom ou Google Meet, tarefas são controladas e monitoradas em "lousas" virtuais pelos gerentes de projeto, e a comunicação acontece de forma mais ou menos assíncrona.

Porém, no início de 2020, o mundo todo foi obrigado a se adaptar a esta nova realidade por conta do novo coronavírus. Grandes empresas tiveram que enviar seus funcionários para casa, para que trabalhassem através da internet, e as empresas que já possuíam uma experiência prévia com este modelo de trabalho saíram na frente na corrida pela transformação digital.

Desenvolvimento de software é uma das principais, se não a principal possibilidade de trabalho remoto no mundo. Esta é uma tendência cada vez maior, de desenvolvedores conquistarem oportunidades de trabalho remoto no Brasil e também no exterior, recebendo em moeda estrangeira e maximizando os ganhos. Existem muitas vantagens, como poder trabalhar em empresas com salário médio maior do que de onde você mora, e até mesmo trabalhar para empresas ao redor do mundo recebendo em moeda estrangeira!



O que todo programador precisa saber

Além dos pontos abordados na seção “**O que todo mundo que desenvolve software tem em comum?**”, existem algumas habilidades técnicas ou interpessoais que costumam ser requisitos comuns nas empresas. É importante lembrar que cada área da tecnologia é um mundo à parte, com suas próprias habilidades necessárias, porém, existem muitas coisas em comum entre elas. Veja algumas delas:

- **Informática básica:** É implícito em toda e qualquer vaga de programação que é esperado que o candidato saiba como instalar e desinstalar programas, como navegar na internet, alterar configurações de seu computador, navegar em arquivos e por aí vai.
- **Pesquisas na internet:** Uma das habilidades mais importantes para qualquer profissional de tecnologia, senão a mais importante. Com a quantidade imensa de ferramentas, comandos e padrões que existem na programação, é impossível se lembrar de tudo. Por isso, desenvolvedores passam grande parte de seu dia pesquisando referências e explicações de como fazer algo no Google, em sites especializados como Stack Overflow ou nos manuais de uso de ferramentas específicas. Se você tem facilidade de resolver problemas pesquisando soluções na internet, vai se sentir em casa na área de programação.
- **SCRUM e como fazer parte de times ágeis:** Atualmente, a esmagadora maioria das empresas que possuem programadores utilizam alguma metodologia ágil para gerir suas equipes, como o **SCRUM**¹. Essa habilidade não costuma estar explícita nas descrições de vagas, porém, por mais que seja algo simples de se aprender na prática em seu trabalho, candidatos que já sabem como se portar em uma equipe que usa SCRUM sempre têm a preferência das empresas.
- **Inglês:** Outra habilidade que pode não ser incluída nas descrições de muitas vagas, porém quando presente, faz a diferença. Você não precisa ser fluente em conversação, mas uma boa capacidade de ler, ouvir e compreender inglês com certeza vai te ajudar a evoluir em sua carreira mais rápido. Os comandos das linguagens de programação, ou seja, as palavras chaves que você precisa utilizar para o computador entender o que você o está instruindo a fazer, são todos em inglês. Muitos termos técnicos não possuem tradução (ou se possuem, não costumamos usar as traduções no mercado) e mesmo no Brasil, muitas empresas exigem que o código seja escrito em inglês. E ainda, muitas ferramentas que você precisará utilizar no dia-a-dia terão seus manuais escritos em inglês. As ferramentas de tradução automática estão diminuindo lentamente a necessidade de saber inglês para programar, porém a qualidade das traduções ainda não é boa o suficiente para retirar completamente a necessidade de um conhecimento básico em inglês.

1. <https://www.atlassian.com/br/agile/scrum>

• **Raciocínio lógico:** Grande parte da programação envolve construir estruturas lógicas. Pessoas com a capacidade de raciocínio lógico mais desenvolvidas terão mais facilidade para aprender a programar. Porém, nada impede que esta habilidade seja adquirida e aperfeiçoada na prática. O próprio estudo da programação vai te ajudar a desenvolver seu raciocínio lógico.

• **Algoritmos e estruturas de dados:** Importantes para programar desde um jogo eletrônico 3D até um aplicativo de loja virtual para smartphones, passando por um braço robótico em uma linha de produção automatizada. Algoritmos são sequências de instruções que precisamos passar ao computador para que ele faça o que queremos, enquanto que estruturas de dados são os “formatos” comuns que os computadores costumam armazenar e representar informação para nós humanos. Novamente, não é necessário estudar estes tópicos de forma isolada antes de buscar uma formação em programação, pois a maioria dos cursos já possuem algoritmos e estruturas de dados básicos em seus currículos. Porém, uma atenção especial a esses temas podem te ajudar a aprender novas tecnologias muito mais rápido no futuro, uma vez que todas as linguagens de programação são baseadas em algoritmos e estruturas de dados.

• **Redes de computadores:** Por mais que não seja o foco de um programador, é importante possuir um conhecimento inicial em redes de computadores para poder construir aplicativos completos, pois essa também é uma habilidade pervasiva na área de tecnologia. A comunicação entre carros autônomos, um jogo online, sites ou um aplicativo de mensagens em tempo real são todos exemplos de programas que dependem de redes para funcionarem.

• **UNIX/Linux:** Estes nomes referem-se a kernels, que são praticamente o coração de um sistema operacional (como o Windows 10 ou o Mac OSX, por exemplo). Mais de 70% da internet é hospedada em servidores executando sistemas operacionais baseados em UNIX, então, para um programador que precisa hospedar seus programas em um servidor, é essencial um conhecimento básico em como lidar com estes tipos de sistemas operacionais. Como grande parte destes servidores não possuem interface gráfica, também é importante conhecer os **comandos bash¹**, que são a forma original de interagir com computadores antes da invenção das interfaces gráficas.

● Planos de estudo (roadmaps)

É comum ouvir o termo “roadmap” quando estamos pesquisando sobre o que estudar em programação. Esse termo se refere a um tipo de diagrama usado para explicar quais habilidades uma pessoa precisa aprender para poder exercer um cargo de desenvolvedor, e qual a melhor sequência para aprender estas habilidades. Abaixo, desenvolvemos alguns roadmaps das principais áreas dentro do desenvolvimento de software para que você possa estruturar um plano de estudos que o ajude a conseguir sua primeira oportunidade na área desejada.



Desenvolvimento Fullstack

[ACESSAR ROADMAP](#)

Considerado o cargo na área de tecnologia **mais procurado por empresas²**, esse profissional vai desempenhar tarefas em todo o espectro do projeto, desde as mais lógicas, como lidar com bancos de dados ou servidores, quanto na parte visual, como construir telas para websites e aplicativos. Mas, o que significa Full Stack? Full, em inglês, significa completo. Já Stack é o termo comumente utilizado para denotar um conjunto de tecnologias utilizadas em um projeto, como por exemplo a linguagem de programação ou o tipo de banco de dados. Portanto, o desenvolvedor full stack é aquele que programa em todo o espectro do projeto, fazendo parte do processo do começo ao fim, desde a construção do banco de dados ao que o usuário vê na tela do seu celular ou computador. É um cargo generalista, onde a pessoa precisará estudar muitos temas diferentes, mas que, por sua versatilidade, costuma ser muito bem remunerado.

1. [https://en.wikipedia.org/wiki/Bash_\(Unix_shell\)](https://en.wikipedia.org/wiki/Bash_(Unix_shell))

2. <https://tecnoblog.net/noticias/2021/09/17/dev-full-stack-e-o-cargo-de-ti-mais-procurado-por-empresas-diz-pesquisa/>



Desenvolvimento Frontend

[ACESSAR ROADMAP](#)

Outro profissional sendo muito procurado pelas empresas, o desenvolvedor frontend atua na “frente de loja” das aplicações, ou seja, na parte visual, que tem interação direta com os usuários. Nos últimos anos, este segmento ganhou muita força por conta dos avanços e investimentos na área de experiência do usuário (**UX¹**). O profissional frontend deve dominar conhecimentos como HTML e CSS para construção de sites/aplicativos web, Javascript para interações dinâmicas e frameworks como React ou Vue, além de temas como **acessibilidade²** e **SEO³**. Por mais que esta seja uma área que lida com construção de interfaces de usuário, a maioria das empresas não exige conhecimentos em design, pois os desenvolvedores frontend trabalham em conjunto com profissionais da área de UX como designers UX/UI ou UX writers. Porém, conhecimentos ou experiências prévias com design de interfaces ou design gráfico costumam ser considerados diferenciais para profissionais desta área.



Desenvolvimento Backend

[ACESSAR ROADMAP](#)

Outro setor com **demandas altíssimas⁴** no Brasil, o profissional backend é responsável pela parte “invisível” das aplicações, ou, como nos referimos na área, pelas “regras de negócio”. Armazenamento de dados, envio de e-mails, buscas e sistemas de autenticação são todos exemplos de tarefas que são executadas pelo backend da aplicação. Desenvolvedores que quiserem se especializar em backend devem dominar temas como servidores web, bancos de dados relacionais e não-relacionais e autenticação de usuários. Por ser uma área “não-visual” da aplicação, no sentido que o resultado de todas as operações do backend são em forma de texto, um conhecimento mais refinado em algoritmos e estruturas de dados pode ser útil e te ajudar a escrever códigos mais otimizados.



DevOps

[ACESSAR ROADMAP](#)

Development Operations, termo comumente abreviado para DevOps, significa literalmente “operações de desenvolvimento (de software)”. Também um dos cargos **mais buscados⁵** pelas empresas, o profissional desta área é responsável por garantir que toda a infraestrutura do software funcione sem nenhum bloqueio, desde o desenvolvimento do código no computador dos programadores até a publicação em um servidor em nuvem. Engenheiros DevOps não são programadores propriamente ditos, pois o foco da função é configurar e testar a infraestrutura por trás do software em si (servidores, bancos de dados, redes virtuais, etc). Porém, hoje, a maioria das ferramentas utilizadas para esse fim podem ser automatizadas usando programação. Por este motivo, saber programar permite aos engenheiros DevOps automatizar tarefas repetitivas e acelerar processos que antes eram feitos manualmente. Esta posição exigirá conhecimentos mais aprofundados em redes de computadores e temas como proxies, firewalls e VPNs, além da necessidade de estar habituado a usar sistemas operacionais baseados em Linux e usar a interface de linha de comando (CLI). Certificações de provedores de serviços em nuvem como AWS e Google Cloud também são muito bem vistas pelas empresas.



Desenvolvimento Mobile

[ACESSAR ROADMAP](#)

Em 2021, houve um **aumento de 600%⁶** na procura por desenvolvedores mobile, em relação à 2020. O desenvolvimento focado em dispositivos móveis se divide entre os dois grandes concorrentes deste mercado: os sistemas operacionais Android (Google) e iOS (Apple). Geralmente, desenvolvedores mobile se especializam em somente uma das plataformas, e empresas que desejam ter seus aplicativos disponíveis para os dois sistemas operacionais precisam ter duas equipes de desenvolvimento diferentes. Porém, a popularização de soluções híbridas como React Native e Flutter está trazendo uma nova opção para as empresas, onde um único time consegue desenvolver aplicativos para ambas as plataformas usando a mesma linguagem e framework. Por este motivo, pode ser vantajoso para o desenvolvedor mobile aprender uma solução híbrida antes de buscar uma especialização em Android ou iOS. Os profissionais dessa área têm requisitos parecidos com a área de **frontend**, porém, com as especificidades de smartphones como geolocalização, leitura de digital ou face, câmera, sensores de rotação e luminosidade, entre outras coisas.

1. https://pt.wikipedia.org/wiki/Experi%C3%Aancia_do_usu%C3%A1rio

2. https://developer.mozilla.org/pt-BR/docs/Web/Accessibility/ARIA/Web_applications_and_ARIA_FAQ

3. <https://developers.google.com/search/docs/beginner/seo-starter-guide?hl=pt-br>

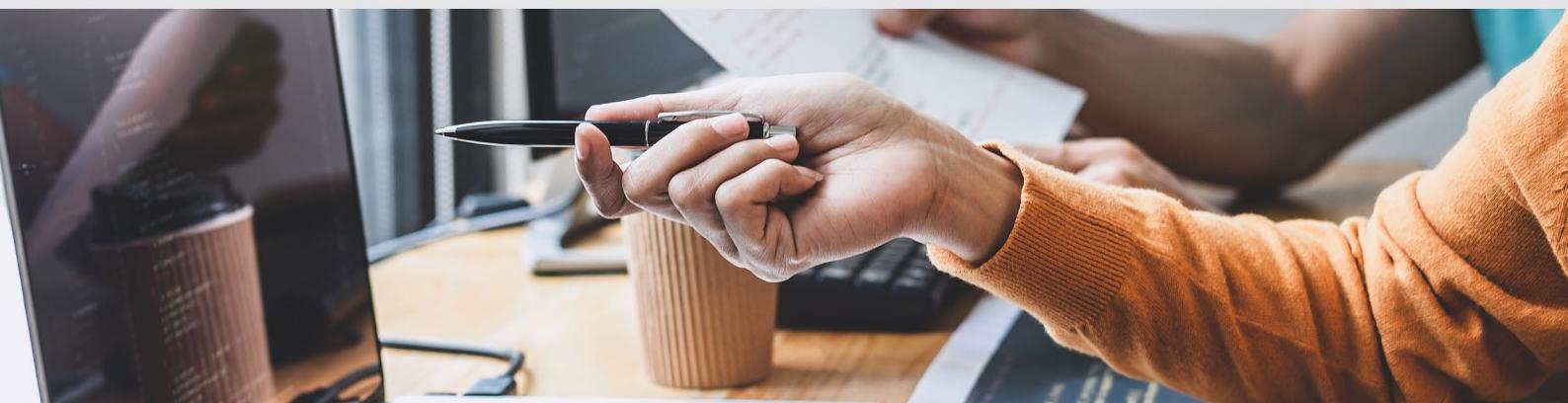
4. <https://olhardigital.com.br/2021/04/29/proalta-demannda-por-desenvolvedores-eleva-media-salarial-em-back-end/>

5. https://blog.geekhunter.com.br/os-profissionais-mais-procurados-no-mercado-de-ti/#Gerenciamento_e_otimizacao

6. <https://tecnoblog.net/noticias/2021/09/29/procura-de-empresas-por-desenvolvedor-mobile-salta-600-em-2021/>

Será que sou realmente capaz?

De tudo que estamos te falando aqui, uma coisa você pode ter certeza: a etapa mais desafiante que existe durante uma carreira profissional em programação para quem está mudando de área é justamente este momento de decisão de buscar uma oportunidade na área e começar os estudos. Assim que você consegue uma oportunidade, um emprego, o segundo maior desafio é o momento inicial do primeiro trabalho na área. Depois, tudo fica muito mais seguro e equilibrado. Por que estamos te falando isso? Pra você desistir? Pelo contrário. Pra te deixar preparado e te dar a tranquilidade de que quando você terminar essa jornada de estudos, conquistar a primeira oportunidade e iniciar sua profissão, tudo será muito melhor emocionalmente.



Aprendendo de verdade e a ilusão de que é incapaz

A maioria das pessoas **não** estão acostumadas a aprender diariamente, ou aprender novos assuntos, em especial, programação de computadores. Muitos têm em mente que aprender é um processo fácil, e, se ficar difícil ou complicado, é porque "não são capazes". Por conta disso, desistem facilmente. Mas é aí que está a questão. Se você não está entendendo um assunto, é neste momento que o aprendizado está para realmente acontecer. É nesta hora que o nosso cérebro realmente começa a "queimar" os neurônios para que a gente consiga processar e entender aquele assunto mais complexo, enxergar padrões e fazer novas conexões.

Durante o seu aprendizado em programação, isso vai acontecer muitas vezes. Será uma rotina divertida, de vencer desafios, e muitas vezes você vai se sentir um "super-herói" quando aquele problema difícil é resolvido com puro código e lógica de programação. É importante se acostumar com isso. Às vezes pode ser frustrante, mas prossiga. É neste momento que devemos persistir e talvez enxergar numa outra ótica. É o momento de pensar "está difícil porque eu ainda não aprendi ou enxerguei uma solução. É agora que eu vou aprender, para tornar o difícil, fácil". É nessa hora que o nosso cérebro realmente estará processando um novo aprendizado. Se acontecer de você desistir, saiba que não é por incapacidade sua, mas muitas vezes por não se dar a chance de realmente tentar aprender um assunto que não conhece.

Portanto, memorize isso: se está difícil de entender ou praticar, é aí que o momento do aprendizado acontece! É nessa hora que você deve persistir até finalmente aprender algo novo de verdade.

Talento é superestimado. Dedicação é o que prevalece!

Ao se deparar com os primeiros obstáculos dentro da programação, você provavelmente vai se questionar se essa área é mesmo pra você. Acredite, **todo** programador já se fez esta mesma pergunta pelo menos alguma vez. Talvez você tenha visto notícias sobre crianças que já sabem programar, ou que Bill Gates vendeu seu primeiro programa de computador com doze anos de idade e pense "eu não tenho talento para isso". Acredite, talento pouco importa nesta área. Estes casos citados acima são exceções, e estando na área, você vai perceber que o processo de aprendizado para a grande maioria das pessoas que desenvolvem software foi mais ou menos o mesmo.

Sim, talento vai facilitar o processo de aprendizagem, porém, não há outra forma de realmente se tornar um bom programador além da prática intensa. A diferença entre as pessoas que conseguem migrar para a carreira de programação com sucesso e as que desistem antes é a persistência. Você terá que escrever milhares e milhares de linhas de código antes de buscar sua primeira oportunidade. Depois que já estiver trabalhando, terá que escrever mais centenas de milhares de linhas de código até poder se considerar proficiente em seu trabalho. A persistência e a disciplina são o que definem o sucesso dentro da programação, portanto, jamais desista por "não ter talento".

Como lutar contra a Síndrome do Impostor



Você já deve ter ouvido o termo "síndrome do impostor". Esse termo se refere à aquele sentimento de "não sou tão bom nisso quanto as outras pessoas próximas a mim, devo ser um impostor entre elas". No contexto do aprendizado de programação, esse sentimento costuma ser uma falácia, ou seja, algo que só é verdade dentro de sua própria cabeça, por dois principais motivos: primeiro, você está automaticamente assumindo que as outras pessoas em seu meio são "melhores" que você baseando-se no que elas falam ou nos sucessos que compartilham. A questão é que a maioria das pessoas não compartilham as dificuldades, somente as vitórias. Pode ter certeza que, quando você compartilhar seus projetos de programação concluídos, outras pessoas que estão começando se sentirão da mesma forma. Além disso, o nível de entendimento de outras pessoas não deveria ser um critério para avaliar sua própria evolução, sendo que cada pessoa possui ritmos e formas de aprendizagem completamente diferentes. O segundo motivo é que você está usando seus próprios critérios para definir o que significa ser ou não um programador. Isso pode parecer um pouco duro, mas a realidade é que não cabe a você decidir se está pronto ou não para começar na área. Quem irá decidir isso serão as empresas que irão te entrevistar, ou os instrutores e mentores que estão te treinando. A grande maioria das vezes, estudantes de programação superestimam a quantidade de conhecimento que precisam adquirir para poder entrar na área. Se realmente fosse necessário saber tanto para trabalhar na área, não teríamos programador algum. Portanto, quando começar a ser "assombrado" pela síndrome do impostor, lembre-se que essa competição é de você contra você mesmo, e que sua única meta é ser um programador melhor hoje do que foi ontem.

Mentalidade de Crescimento

O primeiro contato com programação é infame por nos fazer duvidar de nossa própria inteligência. Porém, é preciso desmistificar o senso comum de que inteligência é algo que temos desde o nascimento ou não. A inteligência, assim como a grande maioria das outras características humanas, é uma habilidade que pode ser aprendida, expandida e aperfeiçoada¹. A programação poderá te fazer se sentir incapaz nas primeiras semanas pois você estará exercitando áreas de sua mente que dificilmente precisavam ser usadas com tanta intensidade antes, mas, quanto mais você aprender, mais fácil será aprender novas coisas. Se pensarmos em nossa inteligência como pensamos em nosso corpo, começar a programar seria mais ou menos como começar a treinar tênis: você estará fortalecendo músculos que nunca foram usados antes, e no início, isso é muito desconfortável. Mas com o tempo e a prática, os treinos vão ficando cada vez mais fáceis, até que você esteja completamente habituado a realizar aquela atividade. Portanto, desligue-se da ideia de que você "não é inteligente o suficiente", pois você constrói sua própria inteligência com muita prática e dedicação. E tenha certeza que as pessoas que você considera inteligentes hoje, provavelmente se sentiram como você anos atrás, porém, passaram milhares de horas praticando programação e hoje precisam de muito menos horas para aprender algo novo relacionado a esse tema. É como se o conhecimento fosse uma grande montanha, porém, quanto mais a escalamos, menos íngreme ela fica. Até que depois de um certo ponto, você deixa de escalar e passa a simplesmente a caminhar. Este é o poder da consistência, perseverança e dedicação.

Expertise

A Curva de Aprendizado

A-ha!
Consegui!!!



Tempo

www.theexcitedwriter.com

1. <https://hbr.org/2016/01/what-having-a-growth-mindset-actually-means>

Veja o que pessoas que começaram uma carreira em programação tem a dizer pra você!

Nada melhor do que receber dicas de quem passou pelo mesmo processo e caminho que você está atualmente! Por isso, separamos abaixo alguns depoimentos de profissionais que começaram do zero uma carreira em desenvolvimento de software. A maioria veio de outras áreas de atuação! Inspire-se com os depoimentos abaixo:



Cecília Sousa

De Jornalismo e Relações Públicas para Frontend Engineer

Tomar coragem para mudar de carreira nunca é uma decisão fácil ou rápida e, no meu caso, não foi diferente. Eu já trabalhava há 9 anos na área de Comunicação quando decidi que queria aprender algo relacionado à tecnologia.

Naquela época, eu pouco entendia o que era backend, frontend e fullstack, mas sabia que seria uma experiência desafiadora e que me tiraria da zona de conforto em que eu estava. Foi assim que comecei a pensar na possibilidade de pivotar minha carreira, mas a certeza só veio depois de conhecer diversas histórias de outras mulheres que tinham passado por esse processo.

Uma dessas histórias foi justamente a de uma ex-aluna da ONG {reprograma} – ela havia deixado sua carreira na área de Letras para atuar como desenvolvedora front-end. Não levou nem um minuto para eu decidir pesquisar por essa instituição, que oferece cursos de desenvolvimento web totalmente gratuito para mulheres cis e trans. Me arrisquei no processo seletivo e, é claro, não passei de primeira. Mas insisti até finalmente ser selecionada para fazer parte de uma turma em 2019.

Eu acredito que o mais importante no meu processo de mudança de carreira foi ouvir depoimentos de outras pessoas, especialmente mulheres, que também tiveram coragem e conseguiram fazer essa transição para a tecnologia dar certo. Sentir essa segurança de que seria possível foi o passo decisivo para mim. E claro que a trajetória em si nunca é uma linha reta: quando o código não funcionava durante as aulas do curso, ou mesmo quando recebi os primeiros "não" para as vagas que me apliquei, batia uma incerteza sobre ter feito a escolha certa. Mas depois de muito persistir e me dedicar, hoje eu olho para trás com a certeza de que estou exatamente onde queria estar!



Guilherme Baldo

De Engenharia Elétrica para Engenheiro de Machine Learning

É muito intimidante pensar em mudar para uma área diferente. A gente nunca se sente preparado o suficiente. Nós achamos que não sabemos tudo, e isso gera um receio, é normal. Na área de desenvolvimento isso é bem explícito, o número de ferramentas e frameworks que precisamos aprender é imenso. Mas é preciso ter em mente que ninguém sabe de tudo e nem é preciso saber de tudo de antemão. Muitos desafios nós só vamos enfrentar na prática e o necessário é saber os fundamentos e ter a vontade de encarar estes desafios. Pensar assim foi crucial para que eu realmente conseguisse mudar de carreira e me tornar um desenvolvedor de software focado em inteligência artificial e machine learning.



Fernanda Moya ••

Senior Android Engineer na Disney Streaming

As pessoas têm medo do início. Às vezes as pessoas querem, mas não priorizam e não executam. Não adianta querer aprender algo novo e não colocar uma prioridade no dia-a-dia. Tem que ser algo rotineiro e consistente, todos os dias. Consistência em programação é chave. Persistência é chave. Isso foi e é muito importante pra mim. Tem que começar aos poucos, nunca será perfeito no começo, pelo contrário. Pelo menos comigo não foi. Desculpa o termo, mas no começo nada fica muito bom, mas depois você vai evoluindo e aprendendo algumas coisas chaves. A principal delas é aprender a pesquisar. Aprender a solucionar um problema por conta própria. Isto foi muito importante pra mim na área de programação. Não que não devemos obter a ajuda de outras pessoas, mas quanto mais a gente aprender a se virar, pesquisar e resolver os desafios, melhor. E repetindo, no início, a coisa mais natural é fazer as coisas meio ruins. Mas aos poucos você vai pegando confiança, construindo. Enfim, esteja pronto pra lidar com desafios que você nunca fez antes. Isso é mágico e muito recompensador em desenvolvimento de software. Programar é isso, desenvolver algo novo, aprender, pesquisar e executar.



Renan Timbó Silva ••

De Bancário para Desenvolvedor Full Stack

Após muita reflexão, planejamento, conversas e conselhos de amigos e da família, decidi realizar uma transição de carreira para a área de tecnologia. Deixei uma carreira de 12 anos em um emprego estável em um banco público, onde era concursado, pelo sonho de seguir na área que descobri ser a minha paixão e sim, estou aqui pra falar que é possível realizar uma transição de carreira saudável após os 30 anos de idade. Foram praticamente 3 anos de planejamento e estudo para, enfim, conseguir finalizar essa mudança. Com isso aprendi algumas coisas importantes no meio do caminho, que gostaria de compartilhar com vocês e que acredito que possa ajudar e até mesmo acelerar esse processo.

Primeiro: é importante organizar os seus estudos pensando se existe algum pré-requisito para cada tópico, para que não pule etapas e perca tempo tendo que voltar e estudar algo que já deveria ser estudado.

Segundo: faça! Coloque a mão na massa, crie algo (foque no simples e feito, evite o complexo que não irá terminar) tente, erre e aprenda. Posso dizer com toda certeza, você vai aprender muito mais fazendo. Estudar e ter a base de conceitos é importante, mas é fazendo que você vai colocar tudo em prática e fixar melhor todo o estudo.

Terceiro: crie um portfólio e coloque todos os seus projetos. Crie uma conta no Github ou qualquer outro site de sua preferência e inclua o link no seu currículo. Sempre que tiver oportunidade, forneça para que as pessoas possam ver o que você sabe fazer. Sem experiência comprovada anterior, é importante mostrar não só o que você sabe fazer, mas que está buscando evolução e desenvolvimento constante.

Enfim, acredito que planejamento e direcionamento são bases fundamentais nesse processo, que dão suporte para que você possa estudar, praticar e se mostrar de forma positiva para as empresas e consiga efetivar sua transição.



Quando e como começar meus estudos?



O primeiro passo para começar na programação deve ser escolher um segmento e uma linguagem de programação. Linguagens de programação são as ferramentas do programador, e são responsáveis por servir como uma espécie de "ponte" entre o código binário, que nada mais são do que os circuitos elétricos que o computador usa para representar dados usando somente zeros e uns, e a linguagem natural, ou seja, aquela que nós humanos nos comunicamos. Como o computador não comprehende linguagem natural por enquanto, e nós humanos não comprehendemos código binário, as linguagens de programação foram criadas para facilitar esta interface entre pessoas e computadores. Ferramentas especializadas chamadas compiladores automaticamente "traduzem" o código que você escreve em uma determinada linguagem para um código binário, que o computador consegue entender.

Além de uma linguagem de programação, você também precisa escolher um segmento para começar seus estudos, como Front End ou DevOps, por exemplo. Essa decisão não precisa ser final e não há problema algum em mudar de ideia no meio do caminho, pois muitas coisas aprendidas em um determinado segmento são aproveitadas nos demais. Na maioria das vezes, a linguagem de programação escolhida vai depender do segmento. Por exemplo, ao escolher Front End, você usará o Javascript, que é a linguagem padrão da web e está presente em todos os navegadores, enquanto em DevOps, será mais importante o conhecimento em Bash, que é a linguagem de scripting dos servidores UNIX.

Não se preocupe muito com escolher o segmento "perfeito". Ao invés de passar meses ponderando entre Front ou Back End, escolha algum dos dois e comece seus estudos. Caso não esteja satisfeito, troque para o outro. É importante curtir o que está aprendendo. Uma boa forma para conhecer melhor o cotidiano de um programador em cada segmento é conversar com pessoas da área. Pergunte a seus amigos ou conhecidos, familiares, ou até mesmo estranhos no Linkedin sobre o que fazem em seu dia a dia trabalhando como programadores de um segmento específico e se imagine na mesma posição. Se pareceu interessante pra você, provavelmente este segmento será uma boa escolha.

É importante também lembrar que dentro do mesmo segmento, diversas linguagens de programação diferentes podem ser utilizadas. Para decidir qual linguagem você irá estudar primeiro, um excelente recurso é a [Stack Overflow Developer Survey](#)¹, uma pesquisa com milhares de desenvolvedores de software sobre diversos aspectos de suas carreiras, linguagens favoritas, salários, entre outras coisas.

segmento. Por exemplo, ao escolher Front End, você usará o Javascript, que é a linguagem padrão da web e está presente em todos os navegadores, enquanto em DevOps, será mais importante o conhecimento em Bash, que é a linguagem de scripting dos servidores UNIX.



1. <https://insights.stackoverflow.com/survey/2021>



Definindo seu “roadmap”

Com segmento e linguagem definidos, chegou a hora de você definir seu roadmap, ou seja, quais habilidades vai estudar, e em qual sequência. Os roadmaps na seção **Planos de estudo (roadmaps)**¹ são um ótimo começo, pois já trazem todas as principais habilidades necessárias para a maioria das vagas no mercado brasileiro (e mundial!) atualmente, organizadas em uma sequência lógica que irá otimizar seu tempo.



Agora é mão na massa!

Com um plano de estudos bem definido, agora só cabe a você segui-lo. É bastante recomendado matricular-se em um curso básico de programação. Cursos oferecerão um aprendizado guiado, em uma sequência lógica que otimizará seu tempo e esforço. A melhor metodologia para aprender programação, é, sem dúvidas, a **Aprendizagem Baseada em Projeto**². Essa forma de aprender sugere mesclar o aprendizado guiado (aulas expositivas) com projetos reais, onde você precisará sozinho construir algum programa totalmente funcional do começo ao fim. Alguns cursos também promovem a colaboração e trabalho em equipe em projetos em grupo, o que pode ser extremamente benéfico para seu início de carreira, uma vez que dentro das empresas, você certamente fará parte de uma equipe. Outro ponto importantíssimo é evitar seguir tutoriais na hora de fazer seus projetos. Tutoriais frequentemente tiram a oportunidade de você praticar habilidades como planejamento e definição de escopo, e muitas vezes, oferecem uma sequência de passos que sempre funcionará de primeira, te impedindo de desenvolver habilidades como tratamento de erros ou debug do seu código. Lembre-se, não caia no **Tutorial Hell**³! Você pode ainda seguir um tutorial para ter uma noção da estrutura de um projeto completo, mas desenvolver um projeto totalmente “seu” em seguida.



Construindo seu portfólio

Construir um bom portfólio é essencial para alguém buscando uma primeira oportunidade na área de tecnologia. Quando você se candidata a uma vaga, a primeira coisa que os recrutadores irão fazer será acessar seu Linkedin e seu perfil no Github, para avaliar se você realmente demonstra ter as habilidades que a vaga exige. Quando você diz saber algo envolvendo programação, é esperado que em seu perfil do Github exista um projeto que “prove” esse conhecimento. Um bom portfólio de programação deve demonstrar várias coisas:

- **Consistência (os famosos “verdinhos” do Github)**
- **Aderência às boas práticas e convenções da linguagem que você usa**
- **Flexibilidade (vários projetos diferentes, de diferentes setores)**
- **Qualidade da documentação (explicar o que seu projeto é, como ele funciona e como é possível contribuir com ele)**
- **Qualidade do código em si (seu código faz o que ele deveria fazer? Você se preocupou com performance? E legibilidade?)**

Muitas vezes, também pode ser vantajoso ter projetos focados na vaga que você mais quer. Por exemplo, ao se candidatar para uma vaga em uma grande empresa de lojas virtuais, você pode desenvolver um pequeno projeto de loja virtual para demonstrar interesse e familiaridade com o tema. No geral, pense que seu portfólio será seu cartão de visitas para os recrutadores, portanto, um perfil organizado, focado e consistente com certeza te trará mais oportunidades.



1. <https://docs.google.com/document/d/1SuEC8u09rHEPMbrGDKZMxSAjpaDsLbWLVYZFcYwm3Y/edit#heading-h.t7292frpegi8>

2. https://en.wikipedia.org/wiki/Project-based_learning

3. <https://javascript.plainenglish.io/tutorial-hell-how-can-you-escape-it-8a6a7da3ae08>



Contribuindo com o Open Source

Open source significa “código-fonte aberto”, e é um movimento que surgiu há algumas décadas atrás como uma maneira de desenvolver softwares de forma colaborativa. Ao contrário do tradicional modelo de código-fonte “fechado”, onde o código fonte do programa é um segredo industrial da empresa que o desenvolveu, e somente uma versão executável do programa, já convertida em código binário é fornecida aos usuários, em programas open-source, o código-fonte costuma ser publicamente acessível através da internet, e qualquer pessoa pode contribuir com ele. Muitos softwares famosos, como o **Linux¹**, **Visual Studio Code²** e **React³** possuem licenças de código aberto, e você pode não só ver todo seu código-fonte, como também modificá-lo à vontade, criar suas próprias versões e customizações deles, e até mesmo sugerir melhorias ou correções nos repositórios oficiais. Contribuir com projetos open source é uma excelente forma de demonstrar conhecimento técnico e domínio sobre ferramentas de controle de versão e colaboração entre desenvolvedores, e costuma ser muito bem recebida pelas empresas, principalmente porque **projeto open source costumam ter uma qualidade de código muito superior⁴** quando comparado com o código desenvolvido em empresas privadas. Ou seja, lá você aprenderá a escrever código de qualidade. Caso você queira começar a contribuir com projetos open source, a iniciativa **Good First Issue** busca trazer tarefas em aberto em projetos open source populares que sejam adequadas para iniciantes na programação.



Qual o caminho é melhor pra mim? Faculdade? Curso online?

Há alguns anos, empresas como Google, Apple e IBM **deixaram de considerar diplomas universitários como pré-requisitos⁵** para suas oportunidades. Esse movimento veio principalmente para acomodar profissionais de tecnologia, principalmente de desenvolvimento de software, que não aprenderam suas habilidades dentro de uma universidade. Com o tempo (e após milhares de contratações), o mercado de tecnologia entendeu que o modelo tradicional de ensino, uma graduação de quatro ou cinco anos acompanhada de um estágio, apoiada em literatura formal, não consegue acompanhar o ritmo em que a tecnologia evolui. Os protocolos e ferramentas que dependemos dentro da área de desenvolvimento de software estão em evolução constante, e qualquer setor está fadado a sofrer uma transformação completa em média a cada cinco anos.

Isso significa que ferramentas consideradas essenciais cinco anos atrás hoje são obsoletas, e ferramentas que hoje são usadas em poucos nichos poderão ser essenciais daqui a cinco anos. Por isso, é muito difícil para as universidades tradicionais planejar um curso superior de quatro anos de duração que não estará obsoleto quando terminar.



1. <https://github.com/torvalds/linux>

2. <https://github.com/microsoft/vscode>

3. <https://github.com/facebook/react>

4. <https://www.zdnet.com/article/coverage-finds-open-source-software-quality-better-than-proprietary-code/#:~:text=Open%20source%20code%20quality%20surpasses%20standard%20for%20good%20quality%20software>

5. <https://goodfirstissue.dev/>



Por conta disso, e também devido à democratização e modernização do acesso à internet, nos últimos anos houve um boom de cursos online relacionados a tecnologia e programação. É possível aprender tudo que um curso superior ensina sem sair de sua casa, pagando consideravelmente menos que uma graduação completa em uma instituição de ensino superior particular, através de vários cursos curtos para cada assunto específico, com a garantia que você estará aprendendo um conteúdo atual e relevante para pelo menos os próximos cinco anos. Isso significa que um diploma universitário é completamente inútil? Bem, não. Muitas instituições já perceberam essa deficiência e voltaram seus esforços para o ensino da ciência da computação mais "pura", com um foco menor em ferramentas que podem ser substituídas a qualquer momento, e um maior em conceitos teóricos que podem ser aproveitados independentemente de ferramentas ou linguagens. Ainda assim, no mercado de trabalho, existe uma demanda maior por profissionais que já estão confortáveis em utilizar as ferramentas necessárias.

Então, qual a melhor opção pra mim? Essa é uma pergunta muito particular, mas, vamos considerar alguns cenários comuns para respondê-la.

Tenho pressa! Quero uma oportunidade logo

Se o seu objetivo de curto prazo é obter as habilidades necessárias para ingressar rápido na área da programação, caso você já seja formado ou tenha algum tempo de carreira em alguma outra área, um curso superior provavelmente não é a melhor opção. Você terá resultados melhores e mais rápidos com cursos livres ou programas intensivos de imersão (bootcamps), onde aprenderá somente as habilidades necessárias e relevantes para o cenário atual de tecnologia. Caso sinta falta de uma base teórica mais aprofundada, atualmente já existem diversos cursos específicos sobre os temas mais elementares da ciência da computação, mas você pode se aprimorar nestas habilidades já trabalhando na área e usufruindo dos altos salários e grande oferta de vagas.

Tenho tempo. Busco uma formação mais longa

Caso você seja alguém mais jovem, que ainda não tenha decidido qual curso superior cursar e nunca trabalhou, ou pretende trabalhar com pesquisa e desenvolvimento na área de tecnologia através da academia, um curso superior pode ser uma boa opção. Nele você aprenderá uma ampla gama de habilidades teóricas e algumas práticas como Cálculo, algoritmos e estruturas de dados, redes de computadores, eletrônica, entre outras coisas. Você também terá mais facilidade para aprender as ferramentas utilizadas no mercado futuramente, uma vez que conhecerá profundamente os mecanismos complexos por trás delas. Mas é importante estar ciente que a maioria dos cursos de graduação não são muito focados em te deixar pronto para o mercado. Por isso, saiba que você precisará correr atrás de formações adicionais à sua graduação para que esteja aprendendo o que o mercado utiliza de tecnologia. Nisso você também pode recorrer a cursos online, vídeos no Youtube, etc.

Quando posso me considerar pronto para conquistar uma 1ª oportunidade?

O mito do "pronto o suficiente"

Uma das dúvidas mais comuns entre estudantes de programação é "quando saberei que estou pronto o suficiente para me candidatar às vagas?". Para a grande maioria das pessoas, a resposta será "você não saberá", e, acredite, isso é algo bom. A maioria das pessoas que estão iniciando em programação costumam supervalorizar os requisitos das vagas ao mesmo tempo que subestimam suas próprias habilidades. Por este motivo, você não é a melhor pessoa para julgar seu conhecimento. Você só saberá se está pronto quando se candidatar à primeira vaga. Caso você conquiste a oportunidade, parabéns! Você estava pronto. Caso contrário, não desanime. As rejeições são **completamente normais**, e são uma ótima oportunidade para você preencher os "furos" em suas habilidades que acarretaram a recusa. Na área da tecnologia, é comum você receber um feedback sobre seu processo seletivo em caso de recusa para que você possa se aprimorar antes de tentar uma próxima candidatura. Use isso ao seu favor, revise os tópicos em que você acredita que poderia ter ido melhor, aprenda novas habilidades que você não possuía e que fizeram falta no processo seletivo, e continue tentando! A busca pelo primeiro "sim" é cansativa e exige disciplina, perseverança e **muita** paciência, mas, tudo vai ter valido a pena quando você finalmente conseguir entrar em uma das melhores, senão a melhor área para se trabalhar no mundo atualmente.

Defina um grande milestone. Planeje e execute até alcançá-lo!

Como dissemos anteriormente, é muito comum o sentimento de dúvida entre iniciantes na programação em relação ao que significa "estar pronto(a)" para entrar no mercado de trabalho. O fato é que não existe um padrão ou "checklist" de tarefas para se fazer antes de começar a buscar uma primeira oportunidade. O que deve definir o momento ideal para iniciar as candidaturas é sua confiança em suas habilidades. Sem estar confiante que você realmente aprendeu todo aquele conteúdo e consegue aplicar estas habilidades na prática, provavelmente você terá dificuldade nas entrevistas e testes técnicos. A melhor maneira de desenvolver habilidades sólidas e acreditar que realmente sabe fazer algo é aplicando estas habilidades em um projeto real. E como todo bom projeto exige planejamento, é importante que você pesquise e se informe de quais habilidades serão necessárias para completar sua ideia de projeto. Os roadmaps da seção **Planos de estudo (roadmaps)**¹ podem auxiliar nisso, porém, pode ser que seu projeto precise de menos habilidades do que as listadas neles, ou habilidades diferentes. Por isso, é importante que você defina qual seria um projeto satisfatório para provar para si mesmo que realmente adquiriu aquele conhecimento.

¹ <https://docs.google.com/document/d/1SuEC8u09rHEPMbrGDKZMxSAjpaDsLbWLVYZFcYwm3Y/edit#heading-h.t7292frpegi8>

O conjunto de habilidades necessárias para concluir este projeto será seu mínimo conhecimento viável: o mínimo que você precisa saber para estar confiante e suficiente para começar a trabalhar na área. Com isso definido, seu trabalho vai ser adquirir todas as habilidades necessárias para fazer o projeto virar realidade, estudando e praticando bastante. Ele será seu *milestone*: o primeiro marco de sua jornada como programador. Uma vez que seu projeto esteja pronto, você deve começar a se candidatar à vagas (ou procurar projetos como freelancer, ou abrir sua empresa) imediatamente, mesmo que ainda reste aquela dúvida em relação a estar pronto. Lembre-se: você definiu que o projeto é a “prova” de seu conhecimento, portanto não há motivos para dúvidas.

Escolher um projeto pode ser desafiador, principalmente para alguém que está iniciando do zero na programação. Veja alguns exemplos que podem ajudar a “ativar” sua imaginação (separados por área de atuação):



Frontend

Projeto: Dashboard para acompanhamento de preços de criptomoedas/ações

Habilidades necessárias:

- HTML/CSS
- Javascript intermediário
- React básico (JSX, state, props, Hooks)
- Roteamento (React Router)
- Interação com APIs (Axios e conceitos básicos de HTTP)
- Manipulação de estruturas de dados (filtros para os dados)
- Publicação e hospedagem de sites (Github Pages, Vercel)
- Git/Github

Backend

Projeto: API REST com CRUD, autenticação e busca avançada (o tema dos dados pode ser qualquer coisa: filmes, músicas, dados sobre a COVID-19, empresas, lugares, etc)

Habilidades necessárias:

- Javascript intermediário
- Programação Orientada à Objetos ou Funcional (escolha uma)
- HTTP (requisições, respostas, verbos, códigos de status)
- Express.js (roteadores, middlewares)
- Autenticação e Autorização (hashing de senhas, autenticação por tokens (JWT) ou por sessões (cookies))
- PostgreSQL (CRUD, relacionamentos, índices e constraints, cláusula JOIN, agregações)
- Sequelize
- Testes de API com Postman
- Publicação e hospedagem de APIs (AWS, Heroku)
- Git/Github

Full Stack

Projeto: Rede social para desenvolvedores com CRUD e autenticação

Habilidades necessárias:

- HTML/CSS
- Javascript intermediário
- React básico (JSX, state, props, Hooks)
- Roteamento (React Router)
- Interação com APIs (Axios e conceitos básicos de HTTP)
- Express.js (roteadores, middlewares)
- Autenticação e Autorização (hashing de senhas, autenticação por tokens JWT)
- PostgreSQL (CRUD, relacionamentos, índices e constraints, cláusula JOIN, agregações)
- Sequelize
- Publicação e hospedagem de APIs (AWS, Heroku)
- Git/Github

Mobile

Projeto: Aplicativo de Reviews de restaurantes com geolocalização

Habilidades necessárias:

- Javascript intermediário
- React básico (JSX, state, props, Hooks)
- React Native
- Interação com APIs (Axios e conceitos básicos de HTTP)
- APIs de geolocalização
- Publicação e hospedagem de apps na Play Store (Android) e App Store (iOS)
- Git/Github

DevOps

Projeto: Publicação de API REST conteinerizada no Kubernetes

Habilidades necessárias:

- Javascript/Node.js básicos
- Pipelines CI/CD (Circle CI)
- Conteineres (Docker)
- Bash básico
- Linux intermediário
- Kubernetes
- Git/Github



Prepare o seu LinkedIn e esteja pronto para as entrevistas

O LinkedIn é uma ferramenta primordial para qualquer pessoa buscando uma carreira em tecnologia. Atualmente já é possível inclusive ser contratado sem sequer enviar um currículo, apenas se candidatando a vagas através do LinkedIn, e tendo um bom perfil que evidencie todas suas habilidades, formação e experiência prévia. Seu LinkedIn vai te ajudar a construir sua marca pessoal e sinalizar as empresas do que você é capaz e com o que pretende trabalhar. É importante que você use o LinkedIn para criar engajamento inclusive **durante** seu aprendizado e não somente quando começar a candidatar-se para as vagas. O LinkedIn é uma rede social, e como todas as outras redes sociais, possui um algoritmo que define quem aparece nas pesquisas. Dessa forma, seu objetivo é maximizar a quantidade de vezes que você aparece nas pesquisas dos recrutadores. Algumas formas de fazer isso são:

- **Fazer publicações:** quanto mais publicações você fizer, maior sua relevância para o algoritmo. Faça publicações sobre os projetos que está trabalhando, sobre os cursos que completou, curiosidades que aprendeu, entre outras coisas. Mostre em vídeo o que você está construindo! ([Veja exemplos¹](#)). Porém, evite o “spam”: suas publicações devem ter algum conteúdo que atraia os recrutadores, caso contrário, fica óbvio que você está apenas publicando para satisfazer o algoritmo.
- **Interagir com publicações de outros usuários:** Curtir, comentar e compartilhar publicações alheias também aumenta sua relevância. Busque sempre interagir com publicações que sejam relacionadas com o que você está estudando. Outra vantagem das interações com outros usuários é que você pode até mesmo criar contatos profissionais para pedir indicações ou referências futuramente!
- **Mandar mensagens:** Sim, até mesmo enviar mensagens afeta sua posição nas pesquisas. Você pode tentar conversar com pessoas que já trabalham na área para tirar dúvidas, pedir conselhos, receber feedback sobre seus projetos, perguntar sobre o cotidiano da pessoa dentro de uma empresa, etc. Você pode também ajudar outras pessoas na mesma situação que você com o que já aprendeu. E não desanime caso não seja respondido por todas as pessoas que contatar: muitas pessoas, principalmente na área de tecnologia, recebem um excesso de mensagens no LinkedIn e para elas é difícil conciliar isso com o trabalho ou estudos.
- **Participar de grupos:** Grupos são a forma que o LinkedIn permite que pessoas compartilhem interesses em comum. Você pode entrar no grupo da linguagem de programação que está aprendendo, grupos de vagas em tecnologia, grupos sobre ferramentas específicas, entre outros.
- **Ter palavras-chave em seu título:** Quando um recrutador está buscando candidatos para uma vaga, ela ou ele fará uso de palavras-chave para encontrar pessoas com experiência nas habilidades necessárias para a vaga. Muitas vezes essas palavras-chave envolvem a linguagem de programação ou ferramentas que a vaga exige. É importante que seu título inclua todas as palavras-chave das tecnologias que você pretende trabalhar com. Por exemplo, caso você esteja buscando uma oportunidade como desenvolvedor Front End em uma empresa que usa React, um bom título seria: “Desenvolvedor Front End | React | Javascript | Typescript | ES7+”.

1. <https://www.linkedin.com/in/gabrieldiasss/recent-activity/shares/>



Com seu perfil ajustado, basta continuar se candidatando às vagas que achar interessante e esperar os contatos dos recrutadores. É importante que você esteja sempre atento às mensagens do LinkedIn, pois muitos recrutadores te contatarão por lá. Instale o aplicativo em seu celular ou habilite as notificações de mensagens por e-mail para não esquecer de responder recrutadores que entrem em contato com você.



O que colocar no LinkedIn se eu não tiver nenhum trabalho anterior?

Caso esteja buscando seu primeiro trabalho, pode parecer que seu LinkedIn parecerá "vazio" aos olhos dos recrutadores, mas é possível criar um bom perfil que chamará a atenção dos recrutadores mesmo sem experiência prévia. Primeiro, já que você não tem experiências para preencher na seção de "Experiência de trabalho", foque nas outras seções de seu perfil. Quais cursos online você já completou? E a faculdade, caso tenha optado por uma? Em sua faculdade, participou de alguma feira ou projeto de extensão? Iniciação científica? Fez parte de alguma empresa júnior? Escreveu artigos? Toda e qualquer experiência é válida, mesmo que "fora" do mercado de trabalho.

Além disso, uma boa ideia também é de você adicionar um cargo chamado "Aspiring Software Developer" (Aspirante a Desenvolvedor de Software). Na descrição deste cargo, é uma excelente oportunidade de falar o que você já aprendeu e todos os projetos que já desenvolveu durante a sua jornada de estudos. Disponibilize também o seu GitHub na descrição!

Caso esteja participando de algum bootcamp ou algum programa de formação de desenvolvedores, você também pode adicionar sua participação como um "cargo de trabalho", detalhando o que você está desenvolvendo no curso e os resultados já atingidos.

Fora isso, você também pode preencher trabalhos voluntários que realizou: que tal renovar o site de uma ONG que você admira? Além de ajudar quem precisa, você está criando uma experiência real que pode ser compartilhada no LinkedIn.

Além dessas coisas, é mais importante ainda que você seja muito ativo na rede. Publique todos os projetos em que está trabalhando, ajude outras pessoas na mesma situação que você, participe de discussões em grupos, interaja com as publicações de sua rede, enfim, é importante que você apareça o máximo possível pois isso otimizará suas chances de uma primeira oportunidade.

Outra ação que pode ser interessante para pessoas sem experiência são as "hackathons": maratonas de código, que geralmente são competições entre equipes com o objetivo de criar um aplicativo ou solução em um curto espaço de tempo. Essa pode ser uma experiência muito rica para o seu perfil, que certamente chamará a atenção dos recrutadores.

Como lidar com a lista infinita de requisitos de uma vaga?

Muitas vezes, as vagas de entrada terão listas enormes de requisitos, e você pode pensar: eu nunca vou conseguir aprender tudo isso. Não se desespere, esse é um fenômeno comum e acontece principalmente por dois motivos:

- **As descrições de vagas não são escritas por pessoas técnicas:** é comum que o papel de definir e escrever requisitos de vagas seja desempenhado por pessoas sem a formação técnica necessária para compreender o motivo da vaga solicitar aquelas habilidades específicas, e muitas vezes, a equipe técnica da empresa não tem tempo de revisar estas informações. Se você possuir a maioria das habilidades principais que a vaga demanda, você já pode se candidatar, pois na maioria das vezes, o que define o seu sucesso nos processos seletivos é sua performance nas entrevistas e testes técnicos e não um "checklist" do que você sabe ou não sabe.
- **A empresa está tentando evitar candidatos muito despreparados:** Ao requisitar muitas habilidades, às vezes muito mais que o realmente necessário para o nível da vaga, a empresa está estrategicamente desencorajando candidatos muito despreparados de congestionarem a caixa de entrada dos recrutadores com candidaturas de baixa qualidade. Isso não significa que você se enquadre nessa categoria. Se você tiver a maioria, ou até mesmo metade dos requisitos, candidate-se e tenha como meta deixar uma impressão positiva nas entrevistas e testes técnicos, e provavelmente a falta de um ou dois requisitos sequer será notada pelos responsáveis pelo processo seletivo.



E lembre-se, diferenciais são diferenciais, não requisitos. Nunca deixe de se candidatar para vagas por não ter alguns ou todos os diferenciais. Se as habilidades estão listadas como diferenciais, significa que provavelmente são usadas raramente no nível da vaga, portanto você pode aprender estas habilidades no próprio trabalho, ou estudando por conta própria após já estar trabalhando. Muitas vezes, para as empresas é mais vantajoso treinar um colaborador cujo caráter e personalidade "encaixam" na cultura corporativa do que optar por um candidato mais forte tecnicamente que provavelmente não vai se adequar ao ambiente. Portanto, não se deixe intimidar por longas listas de requisitos: se você consegue criar projetos sozinho, você consegue trabalhar como programador!



E agora? O que vocês sugerem para o meu próximo passo?



Ok, me convenceram. O que devo fazer?

Primeiramente, parabéns! O fato de você ter lido todo este e-book já é um primeiro passo em direção à sua nova carreira. Esperamos que ele tenha lhe trazido muitos esclarecimentos para este momento. Agora você já têm conhecimento sobre os diferentes segmentos dentro da área de programação, salários médios, dicas para elaborar um plano de estudos efetivo, "armadilhas" que deve evitar, quais habilidades deve ter antes de procurar uma oportunidade, como se portar e adequar sua mentalidade e até mesmo como otimizar suas redes sociais para conseguir uma proposta.

O próximo passo é escolher um segmento (Front End, Back End, Full Stack, Mobile ou DevOps), uma linguagem de programação e um curso online, ou presencial, ou um curso de graduação, tecnólogo. É você quem manda! Só não recomendamos aprender apenas com o Google, ou seja, pesquisando tutoriais e/ou assistindo vídeos no Youtube, pois o seu progresso pode ser bem mais demorado e um tanto caótico. Sim, tem gente que conseguiu uma vaga fazendo apenas isso, mas o caminho é bem mais demorado. Enfim, com isso decidido, mãos à obra! Lembre-se que só se aprende a programar programando, portanto, de hoje em diante, você irá escrever código **todo dia**. Foque também em construir seu portfólio no Github durante seus estudos. Se ainda não conhece, depois você vai entender este termo: "faça commits TODOS OS DIAS!".

Quando você aprender todas as habilidades que definiu como seu *milestone* (lembre-se do que falamos em "Defina um grande milestone. Planeje e execute até alcançá-lo!") e terminar de desenvolver seus projetos, está na hora de se candidatar para as vagas que sejam de seu interesse. Procure por vagas de nível júnior, mas se você tiver todas as habilidades sugeridas em nossos roadmaps, é possível até mesmo se candidatar em algumas vagas de nível pleno. Porém, a maioria das empresas exige experiência em vagas de nível pleno para cima, portanto, leve isso em conta e esteja preparado para defender seu caso nas entrevistas.

Como montar um plano

Não basta só definir qual linguagem, segmento ou primeiro projeto dentro de sua cabeça. Para você realmente se sentir comprometido a começar, é importante colocar tudo isso no “papel”. Papel entre aspas pois hoje temos uma abundância de ferramentas digitais para controle e acompanhamento de tarefas, como o [Notion¹](#) ou o [Trello²](#). O que queremos dizer é que suas metas e objetivos precisam sair de dentro de suas ideias e serem escritas, para que você possa visualizá-las toda vez que se sentar no computador. Mas claro, se preferir o tradicional papel e caneta, fique à vontade. O importante é ter um controle de onde você quer chegar, e por onde já passou.

Caso esteja com dificuldades de estruturar seu plano de estudos, disponibilizamos um exemplo abaixo:

Sugestão de plano de estudos iTuring (Área: Frontend)

The screenshot shows a Trello board titled "Exemplo de Plano de Estudo iTuring". The board has four columns: "A fazer", "Em progresso", "Pendente", and "Concluído".

- A fazer:**
 - Aprender Bootstrap
 - Aprender SASS
 - Aprender sobre deploy no Github Pages
 - Aprender React básico
 - Aprender React intermediário
 - Aprender Styled Components
 - Aprender Material UI
 - Aprender Redux
 - Aprender sobre deploy na Vercel
 - Desenvolver meu segundo projeto - CryptoDashboard
- Em progresso:**
 - Desenvolver meu primeiro projeto - Clone do Dino Game
Progresso: 2/6
 - Aprender Tailwind
- Pendente:** (Empty)
- Concluído:**
 - Aprender Javascript
Progresso: 5/5
 - Aprender HTML e CSS
Progresso: 3/3
 - Aprender Git e Github
Progresso: 5/5
 - Aprender Node.js
Progresso: 5/5

1. <https://www.notion.so/pt-br>
2. <https://trello.com/pt-BR>



Comece agora com o curso gratuito no iTuring!

Bom, se você decidiu começar a estudar programação e optou por um curso online, aqui no iTuring temos um curso 100% gratuito para você dar o pontapé inicial da melhor maneira possível. Reunimos os melhores instrutores e grandes especialistas do mercado para te ensinar a programar do jeito certo.

Baseando-se em tudo que falamos neste guia, o nosso curso gratuito é um pontapé inicial para você avançar no mundo de programação e desenvolvimento de software. É um ponto de partida para uma jornada de sucesso como programador, começando da melhor forma possível. Nosso foco é permitir que você comece do jeito certo, pensando nas coisas certas e a partir daí se desenvolver diariamente. Um dos principais objetivos deste curso é desenvolver em você o hábito de programar todos os dias, pois, sem o hábito, nada do que citamos aqui será útil. Como falamos, lembre-se do que toda pessoa programadora de sucesso tem em comum:

Mentalidade “problem solver”

Prática intensa

Aprendizado constante

Resiliência

Trabalho em equipe

O currículo do curso conta com os fundamentos de ciência da computação e programação de computadores, um tutorial de como usar o git e criar o hábito de ser ativo no Github, além de um guia prático de programação web onde você aprenderá a criar websites e aplicações construindo um projeto do zero! Além disso, você também entenderá como as grandes empresas trabalham com software na prática e como transformar o seu comportamento para se tornar uma pessoa programadora de cada vez mais sucesso.

Nos acompanhe nas redes sociais:

