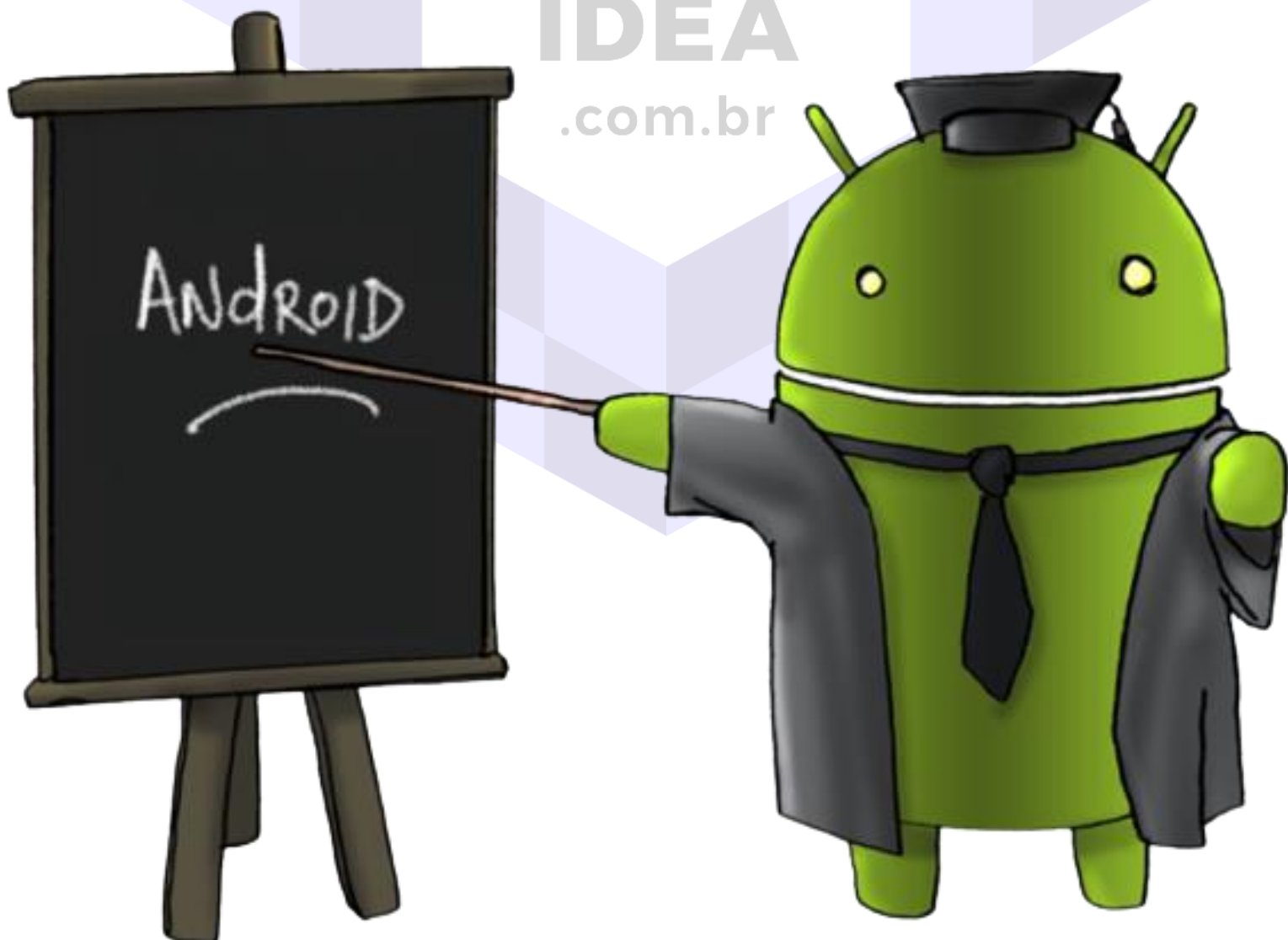


Programação Android

Portal
IDEA
.com.br



História e Evolução do Android

O Android, atualmente conhecido como o sistema operacional mais popular para dispositivos móveis, tem uma história rica que remonta ao início dos anos 2000.

Origem e propriedade do Android

A origem do Android começa com a fundação da empresa Android Inc., em outubro de 2003, por Andy Rubin, Rich Miner, Nick Sears e Chris White. Originalmente, o objetivo do projeto não estava claro para o público, mas especulou-se que o sistema estava destinado a câmeras digitais ou para um sistema operacional de telefones avançado que rivalizaria com os Symbians e Windows Mobile da época.

Contudo, em 2005, o cenário mudou. A gigante da tecnologia Google adquiriu a Android Inc., trazendo Andy Rubin e sua equipe para dentro da empresa. Isso alimentou a especulação de que o Google estava prestes a entrar no mercado de telefonia móvel.

A especulação provou ser verdadeira quando, em 2007, o Google anunciou o Android como um sistema operacional de código aberto para dispositivos móveis baseado no kernel do Linux. A proposta era diferente: em vez de vender o sistema operacional, o Google ofereceria o Android gratuitamente e lucraria com serviços e publicidade. Este modelo de negócios, aliado à flexibilidade e adaptabilidade do Android, atraiu muitos fabricantes, e em 2008 foi lançado o primeiro smartphone com Android, o HTC Dream.

Versões do Android

Desde o seu lançamento, o Android passou por inúmeras atualizações. As versões iniciais (Alpha e Beta) foram seguidas pelo Android 1.0. A partir daí, uma tradição curiosa começou: nomear cada versão do sistema com nomes de doces ou sobremesas, em ordem alfabética.

Aqui estão algumas das versões mais notáveis:

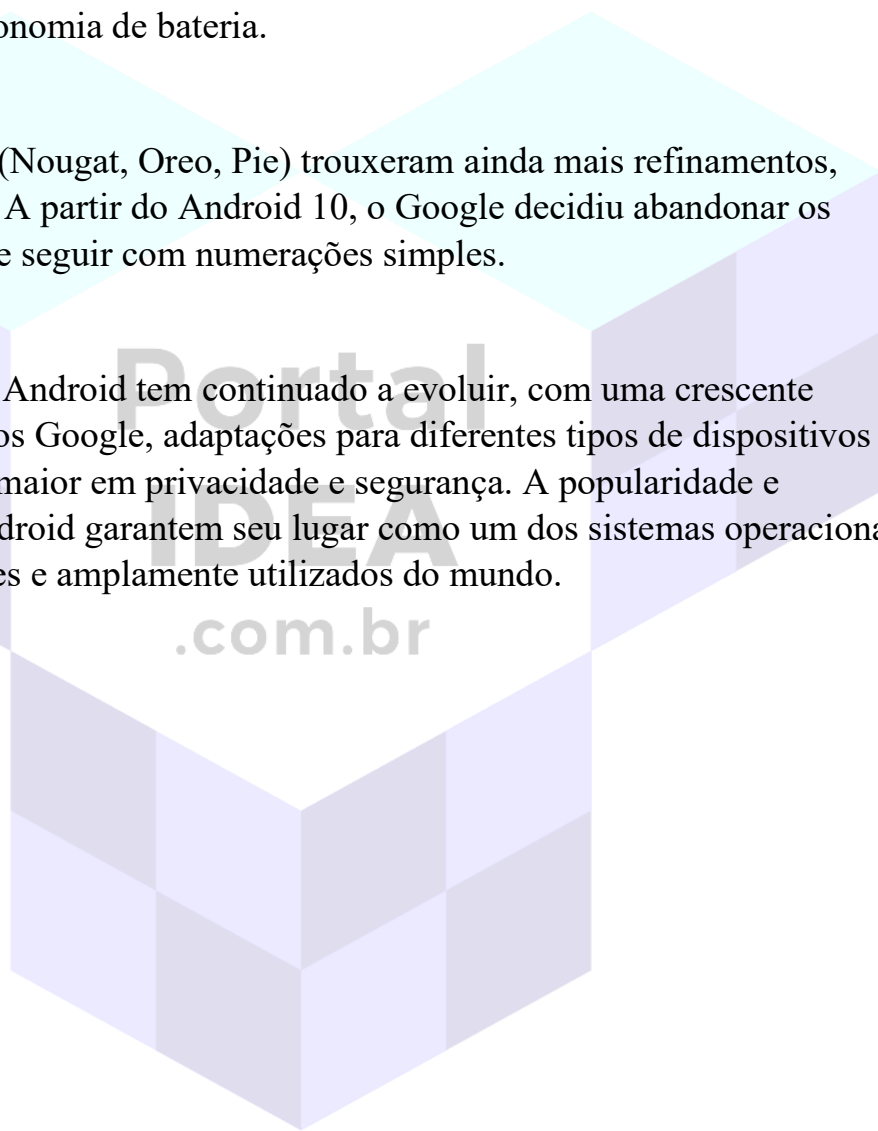
- **Cupcake (1.5):** Introduziu o teclado virtual e a capacidade de gravar e assistir vídeos.
- **Donut (1.6):** Melhorias na pesquisa e na bateria.
- **Eclair (2.0 - 2.1):** Introduziu suporte a várias contas e a câmera flash.
- **Froyo (2.2):** Melhorias de velocidade e o recurso de hotspot móvel.
- **Gingerbread (2.3):** Otimizações para jogos e suporte para câmera frontal.
- **Honeycomb (3.0 - 3.2.6):** Especialmente designado para tablets, introduziu a barra de ação.
- **Ice Cream Sandwich (4.0):** Unificou tablets e smartphones, trouxe reconhecimento facial e melhorias na interface.
- **Jelly Bean (4.1 - 4.3):** Introduziu o Google Now e melhorias na notificação.
- **KitKat (4.4):** Otimização para dispositivos com pouca memória RAM e integração com serviços em nuvem.

- **Lollipop (5.0 - 5.1):** Apresentou o Material Design e várias melhorias na interface.

- **Marshmallow (6.0):** Introduziu o sistema de permissões para aplicativos e o recurso Doze para economia de bateria.

As versões seguintes (Nougat, Oreo, Pie) trouxeram ainda mais refinamentos, segurança e recursos. A partir do Android 10, o Google decidiu abandonar os nomes de sobremesa e seguir com numerações simples.

Ao longo dos anos, o Android tem continuado a evoluir, com uma crescente integração dos serviços Google, adaptações para diferentes tipos de dispositivos e uma ênfase cada vez maior em privacidade e segurança. A popularidade e adaptabilidade do Android garantem seu lugar como um dos sistemas operacionais móveis mais influentes e amplamente utilizados do mundo.



Instalação do Android Studio

O Android Studio é o ambiente de desenvolvimento integrado (IDE) oficial para o desenvolvimento de aplicativos Android. Foi lançado pelo Google em 2013 e, desde então, tornou-se a escolha principal para muitos desenvolvedores de Android devido às suas poderosas funcionalidades e integração direta com o sistema operacional Android. Aqui, exploramos os passos para instalar o Android Studio em seu computador.

Requisitos do sistema

Antes de iniciar a instalação, é crucial garantir que seu sistema atenda aos requisitos mínimos para executar o Android Studio eficientemente:

- Windows:

- **Sistema operacional:** Microsoft Windows 7/8/10 (64 bits)
- **Memória RAM:** 4 GB (8 GB recomendados)
- **Espaço em disco:** 2 GB (4 GB recomendados para Android SDK, emulador e caches de projeto)
- **Resolução de tela:** 1280x800 mínimo

- Mac:

- **Sistema operacional:** macOS 10.10 (Yosemite) ou mais recente, 64 bits
- **Memória RAM e Espaço em disco:** Mesmos requisitos que o Windows

- Linux:

- **Sistema operacional:** Distribuição GNU/Linux, 64 bits
- **Memória RAM e Espaço em disco:** Mesmos requisitos que o Windows

Além disso, para acelerar a emulação do Android, é recomendado ter uma CPU com suporte a virtualização.

Download e instalação

1. Acesse o site oficial: Vá para

[developer.android.com/studio](https://developer.android.com/studio) e clique no botão "Download Android Studio".

2. Aceite os termos: Antes de começar o download, você precisará aceitar os termos de serviço. Uma vez feito isso, o download começará automaticamente.

3. Execute o instalador:

- **Windows:** Execute o arquivo `.exe` e siga as instruções do assistente.
- **Mac:** Abra o arquivo `.dmg` e arraste o Android Studio para a pasta de Aplicativos.
- **Linux:** Extraia o arquivo `.zip` e execute o arquivo `studio.sh` a partir da pasta `bin`.

4. Assistente de configuração: Na primeira execução, o Android Studio apresentará um assistente de configuração que ajudará a instalar o SDK do Android e outros componentes essenciais.

Primeiros passos no IDE

Assim que a instalação estiver concluída e o Android Studio for lançado, você será saudado com uma tela de boas-vindas. Aqui estão alguns passos iniciais a serem considerados:

1. Crie um novo projeto: Selecione "Start a new Android Studio project". Escolha um nome para o seu projeto, configure os parâmetros iniciais e selecione um template para começar.

2. Conheça a interface: Familiarize-se com a IDE. O Android Studio possui diversas janelas, como o Editor de Código, o Logcat (para visualizar logs) e o Design Editor (para layouts).

3. Configure um emulador: Se você não tem um dispositivo Android físico ou simplesmente quer testar seu aplicativo em diferentes dispositivos e versões, pode configurar um emulador Android através do "AVD Manager".

4. Execute seu primeiro app: Depois de criar seu primeiro projeto, clique no botão de "play" (ou pressione Shift + F10) para executar seu aplicativo. Se tudo estiver configurado corretamente, seu aplicativo será compilado e executado no emulador ou no dispositivo conectado.

5. Explore e aprenda: O Android Studio tem muitas funcionalidades, como sugestões de código, refatoração e uma rica integração com bibliotecas e APIs do Google. Dedique algum tempo para explorar e se familiarizar com o ambiente.

O Android Studio é uma ferramenta poderosa que, quando dominada, pode ser a chave para criar aplicativos Android incríveis. Como todo IDE, há uma curva de aprendizado, mas com prática e paciência, ela se tornará uma extensão valiosa do seu processo de desenvolvimento.

Conceitos Básicos do Android

O desenvolvimento Android, embora empolgante, é preenchido por uma variedade de terminologias e conceitos que são vitais para entender e dominar. Entre os muitos conceitos-chave, três se destacam por sua importância e uso frequente: Activity, Ciclo de Vida de uma Activity e Intents.

O que é uma Activity?

Uma `Activity` no Android representa uma única tela com uma interface do usuário. É uma componente fundamental da aplicação que fornece uma janela para a interação do usuário. Por exemplo, um aplicativo de mensagens pode ter uma Activity para a lista de conversas, outra para escrever uma nova mensagem e uma terceira para visualizar as configurações. Cada Activity é independente, mas pode iniciar outras Activities ou ser chamada por elas.

Ciclo de Vida de uma Activity

Cada Activity passa por um ciclo de vida definido, que é uma série de callbacks que o Android chama em diferentes pontos da existência de uma Activity. Dominar esses estados e transições é crucial para criar aplicativos que se comportem de maneira previsível e economizem recursos.

Aqui está um resumo do ciclo de vida de uma Activity:

1. onCreate(): Chamado quando a Activity é criada pela primeira vez. É aqui que você geralmente realiza inicializações, como criar a interface do usuário e preparar alguns dados.

2. onStart(): Este callback é chamado quando a Activity fica visível ao usuário, mas ainda não está em primeiro plano e interagindo com o usuário.

3. onResume(): Agora a Activity está em primeiro plano e o usuário pode interagir com ela. A maior parte do tempo de uma Activity é passado neste estado.

4. onPause(): O sistema chama esse método como o primeiro sinal de que o usuário está deixando a Activity (embora nem sempre signifique que será destruída). Aqui, você deve interromper atualizações e commitar edições ou outras alterações temporárias.

5. onStop(): A Activity não está mais visível para o usuário. Isso pode ocorrer porque outra Activity foi iniciada, ou porque a Activity está sendo destruída.

6. onDestroy(): Este é o último callback e indica que a Activity está prestes a ser completamente destruída.

7. onRestart(): Chamado depois que sua Activity foi interrompida, mas antes do callback `onStart()`.

O que são Intents?

Um `Intent` é uma abstração fundamental no Android e desempenha um papel vital na comunicação entre os componentes da aplicação. Ele age como um mensageiro que solicita uma ação de outro componente, seja ele uma Activity, um serviço ou um provedor de conteúdo.

Existem dois tipos principais de Intents:

1. Intents explícitos: Eles definem o componente que deve responder à intenção. Por exemplo, iniciar uma Activity específica em seu aplicativo.

2. Intents implícitos: Em vez de especificar um componente, eles definem uma ação geral a ser realizada. O sistema, então, interpreta esse Intent e determina qual componente é mais adequado para responder. Por exemplo, abrir uma URL pode desencadear um Intent implícito, e o sistema pode apresentar opções como navegadores disponíveis para lidar com esse Intent.

O desenvolvimento Android gira em torno da compreensão e interação de vários conceitos. Activities, seu ciclo de vida e Intents são apenas a ponta do iceberg, mas são cruciais para criar aplicativos robustos e eficazes. Com uma compreensão sólida desses conceitos, os desenvolvedores podem aproveitar ao máximo o ecossistema Android e criar experiências ricas para os usuários.

.com.br