DESENVOLVIMENTO PARA DISPOSITIVOS MÓVEIS

Diego Bittencourt de Oliveira

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS

Plataformas de desenvolvimento: IDE e emulador Android

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Identificar as plataformas de desenvolvimento e emuladores da Android.
- Descrever as principais características das IDEs atuais e de emuladores.
- Relacionar as principais características entre as plataformas de desenvolvimento e seus emuladores.

Introdução

Neste capítulo, você irá conhecer as principais ferramentas de desenvolvimento de aplicativos para a plataforma Android. Tais ferramentas são capazes de fornecer ao desenvolvedor de aplicativos os recursos necessários para aplicar as funções existentes em um dispositivo móvel, de modo a construir uma aplicação robusta e funcional aos objetivos propostos para esta.

Além disso, você verá as principais características destas ferramentas e aprenderá sobre a IDE da Android (Android Studio) para o desenvolvimento de seus aplicativos , sendo esta a ferramenta indicada pela própria plataforma como a mais adequada para desenvolvê-los.

Você também conhecerá as principais funcionalidades dos emuladores da plataforma Android, que auxiliam o profissional desenvolvedor a testar e a detectar falhas em seus projetos, além de identificar até que ponto estes emuladores conseguem substituir dispositivos reais nos testes de um aplicativo em desenvolvimento.

Ainda neste capítulo, iremos relacionar algumas características do Android Studio com o emulador padrão da plataforma Android, identificando seus recursos e como estes auxiliam no desenvolvimento de um aplicativo livre de falhas.

Plataformas de desenvolvimento e emuladores da Android

IDE, do inglês *integrated development environment*, ou ambiente de desenvolvimento integrado, é um *software* voltado para o desenvolvimento de programas ou aplicativos. O desenvolvedor encontra nos IDEs todas as funções necessárias para o desenvolvimento, desde programas de computador a aplicativos *mobile*, assim como recursos que possibilitam a criação de códigos com uma incidência menor de erros.

No passado, os desenvolvedores necessitavam apenas de um editor de texto e um navegador para criar um *software*. Agora, com os IDEs, uma série de novas opções para otimizar o tempo gasto com códigos foi disponibilizada. Atualmente, os IDEs contam, inclusive, com emuladores capazes de criar ambientes para a realização de testes nos aplicativos desenvolvidos. Assim, é possível realizar modificações no código-fonte e, com poucos cliques, verificar seu funcionamento em um ambiente simulado, ajudando o desenvolvedor a encontrar possíveis falhas de forma mais ágil.

Visando popularizar sua plataforma entre os desenvolvedores, a Android disponibiliza um SDK (*software development kit*) de forma gratuita. Neste SDK, encontramos ferramentas multiplataforma, como o Android Studio e o emulador do ambiente Android.

IDEs de desenvolvimento Android

Atualmente, a Android possui diversos IDEs voltados para o desenvolvimento de aplicativos para sua plataforma. O IDE de maior destaque e o mais indicado para o desenvolvimento na plataforma, pela própria Android, é o Android Studio. No entanto, podemos destacar outros IDEs, como os listados a seguir.

- ***Eclipse*:** foi o primeiro IDE a propiciar o desenvolvimento Android, sendo originalmente criado para ser um IDE para várias linguagens de programação, como PHP, C++ e Java. A Android, por precisar de um IDE para desenvolver aplicativos para a sua plataforma, criou o *plugin* ADT (*Android development tools*), que possibilitou o desenvolvimento Android neste IDE.

- ***NetBeans IDE*:** este IDE se caracteriza pela capacidade de desenvolver aplicativos para *desktop*, dispositivos móveis e Web como Java, JavaScript, HTML5, PHP, C / C ++ e diversas outras linguagens de maneira rápida e fácil. Assim como o Eclipse, utiliza o *plugin* ADT para desenvolver aplicativos para Android.
- ***Visual Studio*:** utilizando o SDK Xamarin.Android, é possível desenvolver aplicativos para Android com a linguagem C#. Outra curiosidade é que um aplicativo desenvolvido neste IDE pode ser facilmente modificado para funcionar em dispositivos que utilizam o sistema operacional iOS.
- ***RAD Studio*:** este IDE de desenvolvimento utiliza a linguagem Delphi ou C++, que possibilita a criação de aplicativos para várias plataformas além da Android, como Windows Phone e iOS, sendo capaz de rodar esse aplicativo em qualquer um destes sistemas operacionais, sem a necessidade de qualquer modificação no código.

O Android Studio, conforme já mencionado, é o IDE indicado e desenvolvido pela própria Android. Foi lançado em 2013, tendo como base o IDE JetBrains Intellij IDEA, sendo este desenvolvido para a confecção de aplicativos em linguagem Java, o que não é uma coincidência, visto que a Android trabalha com esta linguagem de programação de forma nativa no desenvolvimento de seus aplicativos.

A Figura 1 ilustra a visão geral do Android Studio e podemos observar, nos itens a seguir, a interface de vários recursos disponíveis nesta ferramenta.

1. Barra de ferramentas.
2. Barra de navegação. Ajuda na navegação pela estrutura de pastas do projeto.
3. Janela do editor de código-fonte, espaço onde o código-fonte é editado, sendo que esta janela pode se modificar, dependendo do código editado.
4. Barra de janela de ferramentas, onde estão localizados botões para expandir ou ferramentas secundárias.
5. Janela de ferramentas, com o gerenciamento dos projetos, tarefas personalizadas, controle de versão, entre outros.
6. Barra de *status*, que disponibiliza o *status* do projeto, do IDE, mensagens e advertências.

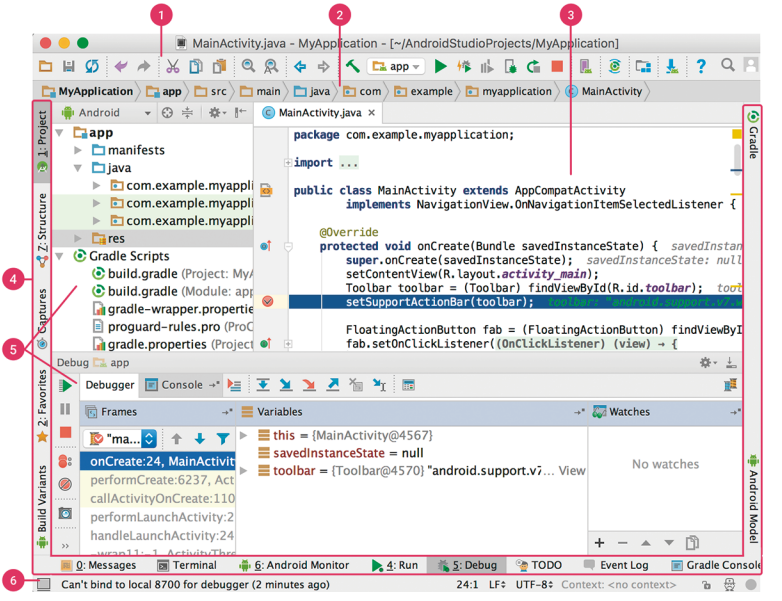

Figura 1. Visão geral da interface do Android Studio.

Fonte: Android Developers (2019a, documento *on-line*).

Link

É possível encontrar mais informações sobre a versão atual do Android Studio, além de opções para realizar o *download* e executar a instalação da ferramenta, no *link* a seguir.

https://qrgo.page.link/Q9CHk

Emuladores do Android

Deitel, Deitel e Deitel (2015) define um emulador como um *software* executado em uma plataforma (por exemplo, Windows, MAC OS X ou Linux), sendo que este deve disponibilizar um ambiente emulado do sistema desejado. Dessa forma, é possível executar as opções deste ambiente, como um aplicativo, e utilizá-lo para verificar seu funcionamento, como uma espécie de ambiente para testes.

No intuito de acelerar o processo de desenvolvimento de um aplicativo Android, a plataforma disponibiliza um emulador multiplataforma em seu SDK, chamado Android Emulator. Por meio deste emulador é exibida uma janela com uma interface realista da Android, conforme observamos na Figura 2.

Figura 2. Visão geral da interface Android Emulator.
Fonte: Android Developers (2019b, documento *on-line*).

Ao criar um AVD (*Android virtual device*, ou dispositivo Android virtual) no Android Emulator, é possível definir todas as características e recursos de um dispositivo tal qual o dispositivo real.

Link

Um AVD consiste em uma configuração na qual as características do dispositivo a ser emulado serão definidas. O AVD possui uma série de configurações que permitem uma grande gama de simulações. No *link* a seguir você pode aprender a realizar estas configurações.

https://qrgo.page.link/AknJ1

Outros emuladores da Android estão disponíveis no mercado. Destacamos alguns a seguir.

- ***BlueStacks*:** emulador Android que permite que o uso de aplicativos Android em um computador, tanto Windows como Mac.
- ***Android x86*:** avançado emulador Android, muito utilizado para jogos e popular entre os desenvolvedores, já que os jogos para Android são executados em computadores sem apresentar quaisquer problemas.
- ***GenyMotion*:** muito utilizado por desenvolvedores, este emulador Android é executado por intermédio do Virtual Box (*software* de virtualização de sistemas operacionais). Oferece um bom desempenho no uso de aplicativos e jogos, oferecendo várias opções de personalização das configurações do emulador.

Link

Você pode encontrar mais informações sobre os emuladores alternativos da Android nos *links* a seguir.

- BlueStacks:

https://qrgo.page.link/mZ4Z6

- Android x86:

https://qrgo.page.link/sisd7

- GenyMotion:

https://qrgo.page.link/XFEsw

Características dos IDEs atuais e emuladores Android

Segundo Simon (2011), o SDK da Android é projetado para permitir que se trabalhe com várias versões da plataforma no mesmo desenvolvimento. Para manter os *downloads* pequenos, a versão do SDK pacotes é separada do SDK. Isso também permite atualizar para novas versões da Android sem ter que baixar o SDK novamente, que pode atingir até mais de 30GB se baixado por completo.

Fique atento

Para uma boa experiência com o Android Studio, você deve estar atento aos requerimentos mínimos de sistema. No caso da plataforma Windows, os requisitos são os listados abaixo.

- Microsoft Windows 7/8/10 32 ou 64 bits.
- 3 GB de memória RAM mínima ou 8 GB de memória RAM recomendado.
- 2 GB de espaço mínimo disponível em disco ou 4GB de espaço disponível em disco recomendados.
- Resolução mínima da tela de 1280 × 800 pixels.

A maioria dos IDEs no mercado atual utiliza o SDK da Android como base para suas aplicações, fornecendo uma ferramenta de criação, gerenciamento e depuração do projeto com base nas funcionalidades fornecidas pelo SDK. O Eclipse e o Netbeans são exemplos de IDEs que utilizam o SDK da Android como base, integrando-o ao *plugin* ADT (*Android development tools*). Logo, não existe uma diferença clara sobre qual dos dois IDEs seria o mais indicado, pois esta é uma questão relacionada ao desenvolvedor. Por exemplo, um desenvolvedor com experiência em aplicativos Java, utilizando o Netbeans, teria uma curva de aprendizado bem menor ao utilizá-lo para desenvolver aplicativos para a Android, visto que já possui experiência com IDEs e conhece suas funcionalidades.

No caso de IDEs como o Visual Studio e o RAD Studio, essa dependência do desenvolvedor vai ainda mais longe. C# e Delphi, linguagens de programação dos IDEs citados, possuem muitos desenvolvedores no mercado. Dessa forma, estas linguagens visam possibilitar aos desenvolvedores, que até então basicamente desenvolviam aplicativos para a plataforma Windows (Windows Desktop, Windows Phone e Web, no caso do Visual Studio), a desenvolver aplicativos para Android, utilizando linguagens de programação nas quais os já possuem experiência. A principal características destes IDEs é a capacidade de criar um projeto Android em suas linguagens nativas (C# ou Delphi) e criar aplicativos Android baseados em Java (ou seja, transformam o código-nativo em Java para, então, o SDK da Android gerar um aplicativo compatível com a plataforma). Este fato pode ser facilmente comprovado, visto que o gerenciador do SDK padrão da Android se encontra disponível nestas ferramentas. A Figura 3 ilustra o aplicativo padrão de gerenciamento das bibliotecas disponíveis no SDK da plataforma.

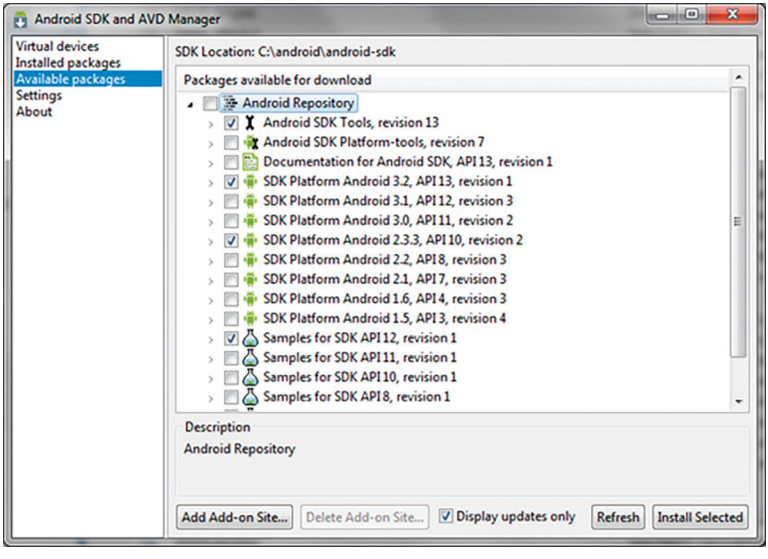

Figura 3. Gerenciador do SDK Android e AVD.

Segundo Deitel, Deitel e Wald (2016), o Android Studio, na condição de IDE mais indicado pela plataforma para o desenvolvimento de seus aplicativos, possui características muitos similares às de outros IDEs, visto que o próprio Android Studio é baseado em um IDE desenvolvido para Java (IDE JetBrains Intellij IDEA, conforme mencionado). No entanto, o Android Studio possui características inseridas pela equipe Android que o tornam uma ferramenta muito completa para a plataforma. Algumas características estão listadas a seguir.

- ***Gradle*:** ferramenta de automatização do processo de compilação do aplicativo.
- ***Layout*:** a interface do IDE permite arrastar componentes para formar o *layout* do aplicativo e editar o código-fonte gerado pela interface. Além disso, permite simular o comportamento do *layout* criado nos diversos tamanhos e formatos de *displays* encontrados nos dispositivos disponíveis do mercado.
- ***Renderização dos layouts*:** função de renderização inteligente de *layout* na qual os erros de *layout* são identificados e sugestões de solução são disponibilizadas ao desenvolvedor.
- ***Análise do código*:** similar a renderização de *layouts*, a análise de código verifica o código-fonte do aplicativo, linha a linha, sugerindo correções ao desenvolvedor.
- ***Repositórios de versionamento*:** possui integração com os principais repositórios de versionamento do mercado, o que possibilita o trabalho em conjunto entre grandes equipes de desenvolvimento, trabalhando simultaneamente em um único projeto.
- ***Plugins*:** apesar de ser uma característica muito comum nos IDEs, o suporte de *plugins* do Android Studio é muito potente, visto que até mesmo o emulador nativo do SDK pode ser substituído pelo uso de *plugins*.

O Gradle é um importante conjunto de ferramentas que automatiza e gerencia o processo de compilação de um aplicativo Android, funcionando como uma espécie de caixa de ferramentas da compilação na qual são fornecidos processos e configurações específicos para compilar e testar um aplicativo Android. Vale observar que o Gradle funciona de forma independente do Android Studio, ou seja, é possível compilar os aplicativos via linha de comando em um computador em que o Android Studio não esteja instalado.

Link

Você pode encontrar mais informações sobre o Gradle em sua página oficial e verificar as informações na página sobre Android, que aborda o processo de compilação de aplicativos.

https://qrgo.page.link/Uvo8Z

O emulador disponível no SDK da Android é o mais completo, pois possui diversas imagens de todas as versões do sistema operacional Android, sendo que por seu AVD é possível criar dispositivos com as mais variadas características e dos mais variados tipos:

- *smartphones*;
- *tablets*;
- relógios inteligentes (dispositivo Wear OS);
- Android TV (sistema operacional Android direcionado a televisores *smart*).

Dentre as características que podemos definir para um dispositivo pelo AVD, se destaca a possibilidade de personalização do *hardware*. Abaixo, temos algumas das personalizações possíveis:

- poder de processamento;
- quantidade de memória RAM;
- dimensões e resolução do *display*;
- botões e tipos de teclado disponíveis;
- sensores (GPS, sensor de proximidade, giroscópios, acelerômetros, entre outros);
- câmeras.

Neste emulador também podemos simular diferentes operações como uma ligação telefônica sendo recebida, um SMS sendo recebido, níveis de bateria e localizações do GPS. Dessa forma, é possível realizar simulações de operações que um aplicativo executa em seu uso real e analisar se o comportamento realizado é o comportamento esperado. Por exemplo, quando um *smartphone* recebe uma ligação telefônica, a prioridade é que esta ligação seja atendida e que nenhum aplicativo se imponha sobre ela. Outro exemplo de simulação seria um usuário em meio a uma ligação telefônica que recebe uma segunda chamada de um aplicativo de comunicação como o Skype. Nesse caso, o ideal é que o Skype identifique que o usuário está ocupado e que no momento não pode atender, não gerando alertas que possam interferir na ligação corrente.

Todas essas simulações e inúmeras outras podem ser realizadas pelo emulador do SDK da Android, porém muitos desenvolvedores relatam dificuldades ao simularem aplicativos mais complexos neste emulador. Para suprir estes casos, muitos desenvolvedores utilizam emuladores de outros fornecedores como o Genymotion, um emulador muito potente que roda em um navegador e, inclusive, possui um *plugin* que o integra ao Android Studio. No entanto, esta é uma ferramenta paga, o que acaba afastando alguns desenvolvedores que preferem utilizar outras técnicas gratuitas para solucionar suas dificuldades.

Link

Muitas vezes, configurar o dispositivo físico para trabalhar de forma integrada ao Android Studio pode ser um processo complexo que desmotiva o desenvolvedor. No *link* a seguir podemos encontrar diversas informações sobre a configuração deste recurso.

https://qrgo.page.link/1BksV

Uma técnica possível seria a utilização de dispositivos reais para realizar as simulações e a depuração dos aplicativos, sendo que pelo SDK da Android podemos utilizar um celular comum, como um dispositivo integrado ao IDE, no qual o aplicativo é instalado e iniciado de forma automatizada, tal qual um AVD. Essa técnica necessita de um dispositivo físico e é, inclusive, uma técnica muito recomendada, considerando que o emulador não é um dispositivo real, por mais que forneça recursos. No dispositivo real, o aplicativo pode apresentar comportamentos indesejáveis que não são detectáveis em simulações.

Características das plataformas de desenvolvimento e os seus emuladores

A capacidade de realizar diversas simulações pela plataforma integrada ao emulador ou dispositivo de teste é uma ferramenta muito potente que o desenvolvedor tem em suas mãos para auxiliá-lo na detecção de problemas. Os problemas citados podem ser relacionados a uma lógica de programa não desenvolvida de forma adequada pelo desenvolvedor ou, até mesmo, um problema relacionado a uma incompatibilidade no sistema operacional.

Exemplo

Em um ambiente móvel, erros podem custar muito caro e trazer muitas inconveniências. Um exemplo é a energia elétrica, considerando que a empresa fornecedora se desloca todos os meses até a residência do cliente para coletar seu consumo de energia. Em diversos casos, o cliente pode residir em locais distantes ou de difícil acesso e, portanto, essa coleta pode gerar um custo alto. Na situação em que essa coleta é realizada com o auxílio de um dispositivo móvel, a ocorrência de uma falha ao coletar essas informações pode gerar prejuízo ao funcionário, que deverá retornar à empresa com o dispositivo para que este seja verificado e, então, retornar à residência do cliente e realizar a coleta de dados.

Na Figura 4 é possível observar um exemplo da capacidade do Android Studio de realizar análises durante as simulações com o aplicativo, sendo que esta figura ilustra a técnica de *break point* (ponto de interrupção). Nesta técnica, o programa é executado normalmente e pausa a sua execução assim que encontrar um ponto de interrupção. A partir deste momento, é possível visualizar dados avançados do código-fonte, com os valores atuais de cada variável declarada no aplicativo. Isto serve para averiguar se o comportamento do *software* até esse ponto foi correto, ou executar o código-fonte linha a linha, a partir deste momento, para analisar se o aplicativo apresentará o comportamento esperado.

Neste recurso, o Android Studio é capaz de realizar a análise linha a linha durante a execução de um aplicativo em um dispositivo real integrado ao IDE, o que o torna ainda mais potente, visto que realizar testes reais de um aplicativo já é vantajoso, porém a capacidade de analisar o código em execução no dispositivo em tempo real é um recurso muito precioso.

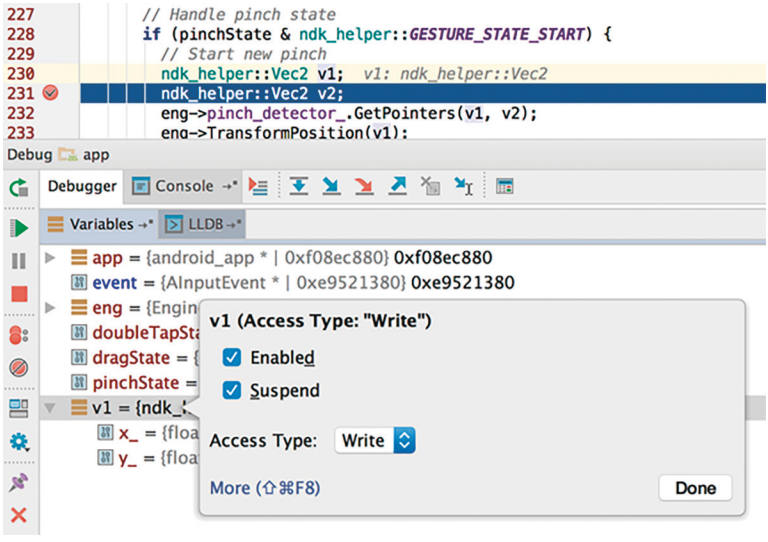

Figura 4. Exemplo de ponto de interrupção no código-fonte do aplicativo e visualização do conteúdo das variáveis.

Fonte: Android Developers (2019c, documento *on-line*).

Segundo Lee, Schneider e Schell (2005), em dispositivos móveis os recursos costumam ser limitados, alguns destes chegando a estados críticos. Observe a seguir.

- **Processamento:** processadores de outros dispositivos são componentes que costumam gerar calor. Em dispositivos móveis, estes componentes possuem boa capacidade de processamento compatível com o dispositivo em questão, porém a emissão de calor pode ser indesejada pelo usuário, logo seu processamento em alta performance deve ser controlado.
- **Memória RAM:** podemos encontrar dispositivos no mercado atual com 8GB de memória, mas em muitos casos não nos depararemos com uma quantidade de recursos de memória tão flexíveis assim. A utilização de memória do aplicativo deve ser controlada, visto que outros aplicativos que convivem no mesmo ambiente também utilizam este recurso.
- **Bateria:** talvez o recurso mais crítico em um dispositivo móvel, considerando que, uma vez consumida a bateria, o dispositivo não pode ser mais operado. O uso consciente do processamento e dos recursos de memória RAM influencia diretamente na quantidade de bateria consumida.

Acima, podemos verificar a importância dos recursos indicados para compreendermos a importância do monitor de recursos do Android Studio, ilustrado na Figura 5. Por meio deste é possível analisar o uso de memória RAM e o uso de processamento do dispositivo móvel enquanto o aplicativo desenvolvido é executado. Este monitor de recursos se torna ainda mais potente aliado ao recurso de *break point* descrito anteriormente, no qual ao executar o aplicativo linha a linha podemos observar a evolução do uso de recursos.

É possível realizar ajustes no aplicativo para que este apresente um funcionamento adequado ao ambiente em que está sendo executado; por exemplo, não consumindo muitos recursos, ou em casos específicos em que consome os recursos necessários, apenas pelo período de tempo necessário à função executada, liberando-os em seguida, visando preservar seu bom funcionamento em um ambiente com diversos outros aplicativos sendo executados de forma concorrente.

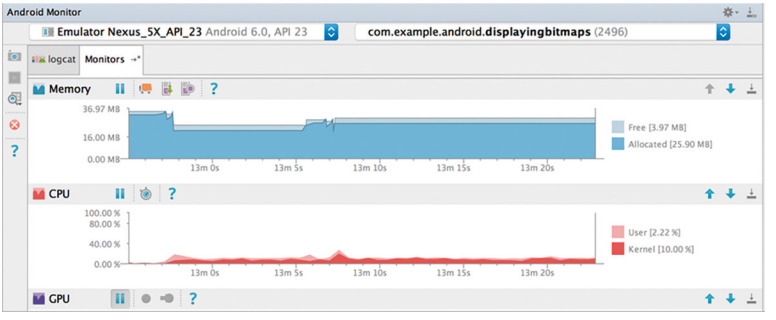

Figura 5. Exemplo de monitoração dos recursos em uso do AVD.
Fonte: Android Developers (2019c, documento *on-line*).

Link

Muitos dos erros de um aplicativo são complexos de se identificar, sendo necessárias técnicas de análise de aplicativo durante seu funcionamento para identificar certos comportamentos. O *link* a seguir fornece informações sobre estas técnicas de depuração no IDE Android Studio.

https://qrgo.page.link/3ibdE

Referências

ANDROID DEVELOPERS. *Conheça o Android Studio*. 2019a. Disponível em: https://developer.android.com/studio/intro?hl=pt-br. Acesso em: 1 jun. 2019.

ANDROID DEVELOPERS. *Depure seu aplicativo*. 2019c. Disponível em: https://developer.android.com/studio/run/emulator.html. Acesso em: 1 jun. 2019.

ANDROID DEVELOPERS. *Executar aplicativos no Android Emulator*. 2019b. Disponível em: https://developer.android.com/studio/run/emulator.html. Acesso em: 1 jun. 2019.

DEITEL, P.; DEITEL, H.; WALD, A. *Android 6 para programadores*: uma abordagem baseada em aplicativos. 3. ed. Porto Alegre: Bookman, 2016.

DEITEL, P.; DEITEL, H.; DEITEL, A. *Android*: como programar. 2. ed. Porto Alegre: Bookman, 2015.

LEE, V.; SCHNEIDER, H.; SCHELL, R. *Aplicações móveis*: arquitetura, projeto e desenvolvimento. São Paulo: Person, 2005.

SIMON, J. *Head first android development*. Farnham: O'Reilly, 2011.

Leituras recomendadas

ANDROID DEVELOPERS. *Configurar sua compilação*. 2019. Disponível em: https://developer.android.com/studio/build. Acesso em: 1 jun. 2019.

ANDROID DEVELOPERS. *Criar e gerenciar dispositivos virtuais*. 2019. Disponível em: https://developer.android.com/studio/run/managing-avds.html. Acesso em: 1 jun. 2019.

ANDROID DEVELOPERS. *Executar aplicativos em um dispositivo de hardware*. 2019. Disponível em: https://developer.android.com/studio/run/device?hl=pt-br. Acesso em: 1 jun. 2019.

ANDROID DEVELOPERS. [*Site*]. 2019. Disponível em: https://developer.android.com/studio. Acesso em: 1 jun. 2019.

Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo:

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS