Módulo 5

Desenvolvimento de Aplicações Móveis

Lição 1

Introdução

Versão 1.0 - Set/2007

Autor
xxx

Equipe
Rommel Feria
John Paul Petines

Necessidades para os Exercícios

Sistemas Operacionais Suportados

NetBeans IDE 5.5 para os seguintes sistemas operacionais:

- Microsoft Windows XP Profissional SP2 ou superior
- Mac OS X 10.4.5 ou superior
- Red Hat Fedora Core 3
- Solaris™ 10 Operating System (SPARC® e x86/x64 Platform Edition)

NetBeans Enterprise Pack, poderá ser executado nas seguintes plataformas:

- Microsoft Windows 2000 Profissional SP4
- Solaris™ 8 OS (SPARC e x86/x64 Platform Edition) e Solaris 9 OS (SPARC e x86/x64 Platform Edition)
- Várias outras distribuições Linux

Configuração Mínima de Hardware

Nota: IDE NetBeans com resolução de tela em 1024x768 pixel

Sistema Operacional	Processador	Memória	HD Livre
Microsoft Windows	500 MHz Intel Pentium III workstation ou equivalente	512 MB	850 MB
Linux	500 MHz Intel Pentium III workstation ou equivalente	512 MB	450 MB
Solaris OS (SPARC)	UltraSPARC II 450 MHz	512 MB	450 MB
Solaris OS (x86/x64 Platform Edition)	AMD Opteron 100 Série 1.8 GHz	512 MB	450 MB
Mac OS X	PowerPC G4	512 MB	450 MB

Configuração Recomendada de Hardware

Sistema Operacional	Processador	Memória	HD Livre
Microsoft Windows	1.4 GHz Intel Pentium III workstation ou equivalente	1 GB	1 GB
Linux	1.4 GHz Intel Pentium III workstation ou equivalente	1 GB	850 MB
Solaris OS (SPARC)	UltraSPARC IIIi 1 GHz	1 GB	850 MB
Solaris OS (x86/x64 Platform Edition)	AMD Opteron 100 Series 1.8 GHz	1 GB	850 MB
Mac OS X	PowerPC G5	1 GB	850 MB

Requerimentos de Software

NetBeans Enterprise Pack 5.5 executando sobre Java 2 Platform Standard Edition Development Kit 5.0 ou superior (JDK 5.0, versão 1.5.0_01 ou superior), contemplando a Java Runtime Environment, ferramentas de desenvolvimento para compilar, depurar, e executar aplicações escritas em linguagem Java. Sun Java System Application Server Platform Edition 9.

- Para **Solaris**, **Windows**, e **Linux**, os arquivos da JDK podem ser obtidos para sua plataforma em http://java.sun.com/j2se/1.5.0/download.html
- Para **Mac OS X**, Java 2 Plataform Standard Edition (J2SE) 5.0 Release 4, pode ser obtida diretamente da Apple's Developer Connection, no endereço: http://developer.apple.com/java (é necessário registrar o download da JDK).

Para mais informações: http://www.netbeans.org/community/releases/55/relnotes.html

Colaboradores que auxiliaram no processo de tradução e revisão

Aécio Júnior
Alexandre Mori
Alexis da Rocha Silva
Allan Souza Nunes
Allan Wojcik da Silva
Anderson Moreira Paiva
Andre Neves de Amorim
Angelo de Oliveira
Antonio Jose R. Alves Ramos
Aurélio Soares Neto
Bruno da Silva Bonfim
Carlos Fernando Gonçalves
Denis Mitsuo Nakasaki
Fábio Bombonato
Fabrício Ribeiro Brigagão
Francisco das Chagas
Frederico Dubiel
Herivelto Gabriel dos Santos
Jacqueline Susann Barbosa
João Vianney Barrozo Costa
Kefreen Ryenz Batista Lacerda
Kleberth Bezerra G. dos Santos
Leandro Silva de Morais
Leonardo Ribas Segala
Lucas Vinícius Bibiano Thomé
Luciana Rocha de Oliveira
Luiz Fernandes de Oliveira Junior
Marco Aurélio Martins Bessa
Maria Carolina Ferreira da Silva
Massimiliano Giroldi
Mauro Cardoso Mortoni
Paulo Afonso Corrêa
Paulo Oliveira Sampaio Reis
Pedro Henrique Pereira de Andrade
Ronie Dotzlaw
Seire Pareja
Sergio Terzella
Vanessa dos Santos Almeida
Robson Alves Macêdo

Auxiliadores especiais

Revisão Geral do texto para os seguintes Países:

- **Brasil** – Tiago Flach
- **Guiné Bissau** – Alfredo Cá, Bunene Sisse e Buon Olossato Quebi – ONG Asas de Socorro

Coordenação do DFJUG

- **Daniel deOliveira** – JUGLeader responsável pelos acordos de parcerias
- **Luci Campos -** Idealizadora do DFJUG responsável pelo apoio social
- **Fernando Anselmo -** Coordenador responsável pelo processo de tradução e revisão, disponibilização dos materiais e inserção de novos módulos
- **Rodrigo Nunes -** Coordenador responsável pela parte multimídia
- **Sérgio Gomes Veloso -** Coordenador responsável pelo ambiente JEDI™ (Moodle)

Agradecimento Especial

John Paul Petines – Criador da Iniciativa JEDI™
Rommel Feria – Criador da Iniciativa JEDI™

1. Objetivos

Nesta lição, discutiremos as características dos dispositivos móveis e a forma de iniciar o desenvolvimento de aplicações para estes dispositivos. Realizaremos uma introdução à *Java Platform, Micro Edition* (Java ME) incluindo a importância das configurações e perfis.

Ao final desta lição, o estudante será capaz de:

- Identificar as características dos dispositivos móveis
- Descrever a arquitetura *JME*
- Conhecer a personalização das configurações e perfis (CLDC e CDC)
- Identificar as bibliotecas fornecidas pelo *MIDP*
- Descrever o ciclo de vida de um *MIDlet*

2. Dispositivos Móveis

Dispositivos móveis podem variar em tamanho, projeto e *layout,* mas eles possuem algumas características em comum que são totalmente diferentes de sistemas *desktop*.

- **Pequenos em tamanho**

  Dispositivos móveis são pequenos em tamanho. Consumidores desejam dispositivos pequenos pela mobilidade e conveniência.

- **Memória Limitada**

  Dispositivos móveis também possuem pouca memória, tanto primária (RAM) quanto secundária (disco). Esta limitação é um dos fatores que afetam a escrita de classes para estes tipos de dispositivos. Com quantidade limitada de memória, devemos fazer considerações especiais acerca da conservação no uso de recursos preciosos.

- **Poder de processamento limitado**

  Sistemas móveis não são poderosos como são os sistemas *desktop* quanto a sua organização. Tamanho, tecnologia e orçamento são alguns dos fatores que influenciam a condição desses recursos. Como o disco de armazenamento e RAM, apenas pequenos pacotes se adequam a estes recursos.

- **Baixo consumo de energia**

  Dispositivos móveis possuem baixo consumo de energia em relação às máquinas *desktop*. Estes dispositivos necessitam poupar o uso de energia, pois possuem um limitado abastecimento através de baterias.

- **Robusto e confiável**

  Por serem dispositivos móveis provavelmente serão carregados. Precisam ser robustos o suficiente para suportarem a força de impacto, movimento e ocasionalmente quedas.

- **Conectividade limitada**

  Dispositivos móveis têm baixa largura de banda, alguns deles não suportam conexão. Outros destes usam conexões de rede sem fio.

- **Curto tempo de inicialização**

  Estes dispositivos inicializam-se em segundos. Tomemos o caso de telefones móveis: eles se iniciam em segundos e as pessoas não ficam com estes desligados mesmo à noite. PDAs inicializam no segundo em que é pressionado o botão de ligar.

3. Visão sobre a Java ME

3.1. Plataforma Java

Java foi criado em 1991 por *James Gosling,* da *Sun Microsystems*. Inicialmente chamada de OAK, em homenagem a árvore que ficava do lado de fora vista da janela de *Gosling*, este nome foi modificado para Java porque já existia uma linguagem chamada OAK.

A motivação original para Java estava na necessidade para uma linguagem independente de plataforma que fosse embarcada em vários produtos eletrônicos de consumo como torradeiras e refrigeradores. Um dos primeiros projetos desenvolvidos usando Java foi um controle remoto pessoal chamado de *Star 7*.

Nessa mesma direção, e ao mesmo tempo, a *World Wide Web* e a Internet estavam ganhando popularidade. Gosling tornava Java capaz de ser usada para a programação para Internet.

Com o lançamento da versão 1.2, a plataforma Java foi classificada em várias plataformas: *Java Standard Edition* (Java SE), *Java Enterprise Edition* (Java EE), *Java Micro Edition* (Java ME) e *Java Card API*.

Java SE – Java Platform, Standard Edition	aplicações *desktop*
Java EE – Java Platform, Enterprise Edition	aplicações corporativas com ênfase no modelo de desenvolvimento *server-side* incluindo *servlets*, JSP, EJB e XML
Java ME – Java Platform, Micro Edition	móveis e dispositivos de mão
JavaCard	Cartões com chip

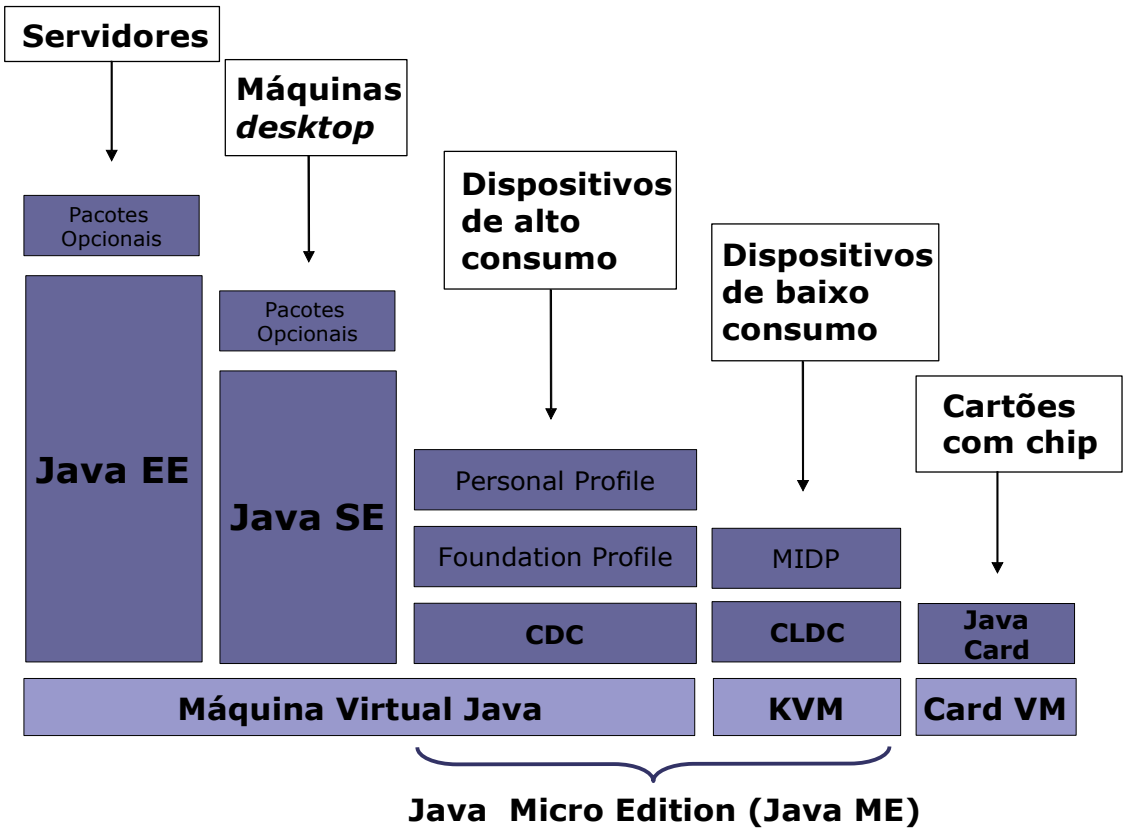

Figura 1: A plataforma Java

3.2 Visão Geral do JME

A Plataforma *Java Micro Edition* (Java ME) é um conjunto de especificações e tecnologias que têm o foco em dispositivos pessoais. Estes dispositivos têm uma quantidade limitada de memória, menor poder de processamento, pequenas telas e baixa velocidade de conexão.

Com a proliferação dos dispositivos pessoais, desde telefones, PDAs, videogames portáteis a aplicações domésticas, Java fornece um único ambiente portátil de desenvolvimento e execução destes dispositivos.

Classes JME, assim como todas as classes Java, são interpretadas. Elas são compiladas em *byte codes* e interpretadas por uma Máquina Virtual Java (JVM). Isto significa que estas classes não são afetadas pelas peculiaridades de cada dispositivo. O JME fornece uma interface consistente com os dispositivos. As aplicações não têm que ser recompiladas para poderem ser executadas em diferentes aparelhos.

No núcleo do JME estão uma configuração e perfis. Uma configuração define um ambiente de execução básico para um sistema JME. Isto define as características das bibliotecas principais, da máquina virtual, segurança e comunicação em rede.

Figura 2: Arquitetura do JME

Um perfil adiciona uma biblioteca para certas classes de dispositivos. Os perfis fornecem bibliotecas da API de interface com o usuário, persistência e mensagens, entre outras.

Um conjunto ou pacote opcional de bibliotecas que fornecem classes funcionais adicionais. A inclusão destes pacotes no JME pode variar porque depende da capacidade do dispositivo. Por exemplo, alguns dispositivos MIDP não possuem *Bluetooth*, logo as APIs de *Bluetooth* não são incluídas nestes dispositivos.

3.3 Configuração

Uma configuração define características mínimas de um ambiente de execução Java completo. Para garantir ótima portabilidade e interoperabilidade entre vários tipos de requisitos de recursos de dispositivos (restrições de memória, processador e conexão), as configurações não contêm as mesmas características opcionais. Uma configuração JME define um complemento mínimo da tecnologia Java. Ela baseia-se nos perfis para definir bibliotecas adicionais (opções possíveis) para uma determinada categoria de dispositivo.

Uma configuração define:

- o subconjunto da linguagem de programação Java
- a funcionalidade da Máquina Virtual Java (JVM)
- bibliotecas do núcleo da plataforma
- características de segurança e comunicação em rede

3.4 Perfis

Um perfil define um conjunto adicional de bibliotecas e características de empresas, de categoria, de dispositivo ou de indústria. Enquanto uma configuração define uma base de bibliotecas, perfis definem as bibliotecas que são importantes para construir aplicações efetivas. Estas bibliotecas incluem a interface com o usuário, comunicação em rede e classes de armazenamento.

4. CLDC

A Configuração de Dispositivos de Conexão Limitada (*Connected Limited Device Configuration* – CLDC) define e endereça as seguintes áreas:

- Características da linguagem Java e Máquina Virtual (JVM)
- Bibliotecas de núcleo (java.lang.*, java.util.*)
- *Input*/*Output* (java.io.*)
- Segurança
- Comunicação em rede
- Internacionalização

4.1. Características Removidas

Algumas características do JSE que foram removidas do CLDC:

- finalização de instâncias de classes
- exceções assíncronas
- algumas classes de erros
- carregadores de classes definidas pelo usuário
- reflexão
- *Java Native Interface* (JNI)
- grupos de processos e processos *daemon*

Reflexão, *Java Native Interface* (JNI) e carregadores de classes definidas pelo usuário são potenciais falhas de segurança. JNI exigem muita memória e pode não ser suportada por dispositivos móveis de pouca memória.

4.2. Características dos Dispositivos CLDC

Os dispositivos atingidos pelo CLDC possuem estas características:

- no mínimo 192kb de memória para a plataforma Java (160kb de memória não-volátil para Máquina Virtual e bibliotecas e 32kb de memória volátil para execução da Máquina Virtual)
- processador de 16 ou 32 *bits*
- baixo consumo de energia (normalmente os que utilizam baterias)
- conexão limitada ou intermitente com velocidade também limitada (normalmente *wireless*)

A CLDC não define instalação da aplicação e ciclo de vida, interfaces com o usuário e tratamento de eventos. Está para os perfis abaixo da CLDC definir estas áreas. Em particular, a especificação MIDP é que define uma aplicação de MIDP (*MIDlet*) que possui um ciclo de vida, biblioteca Gráfica e controle de eventos (classes *javax.microedition.lcdui.**).

4.3. Verificação de Classe

A especificação CLDC exige que todas as classes passem por um processo de verificação de duas fases. A primeira verificação (pré-verificação) deverá está terminada antes da instalação no dispositivo. A segunda verificação ocorre no dispositivo durante a execução, realizada pela Máquina Virtual Java (JVM).

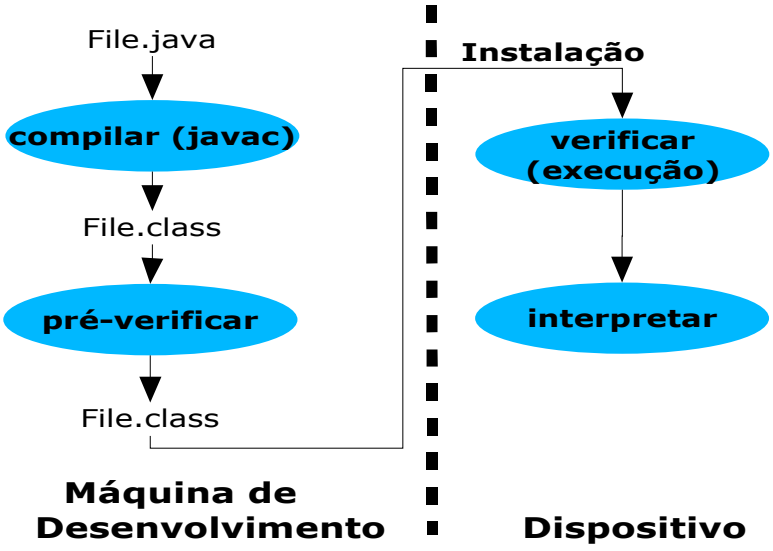

Figura 3: Processo de Verificação em duas fases

4.4. O Framework Genérico de Conexão (GCF)

O *Framework* Genérico de Conexão fornece as APIs básicas para conexão em CLDC. Este *framework* fornece uma base comum para conexões como HTTP, *Sockets* e Datagramas. O GCF fornece um conjunto genérico e comum de APIs que abstraem todos os tipos de conexão. Note-se que nem todos os tipos de conexões são exigidas para serem implementados em dispositivos MIDP.

A hierarquia de interface extensível do GCF torna a generalização possível. Novos tipos de conexões podem ser adicionados neste *framework* através de extensões desta hierarquia.

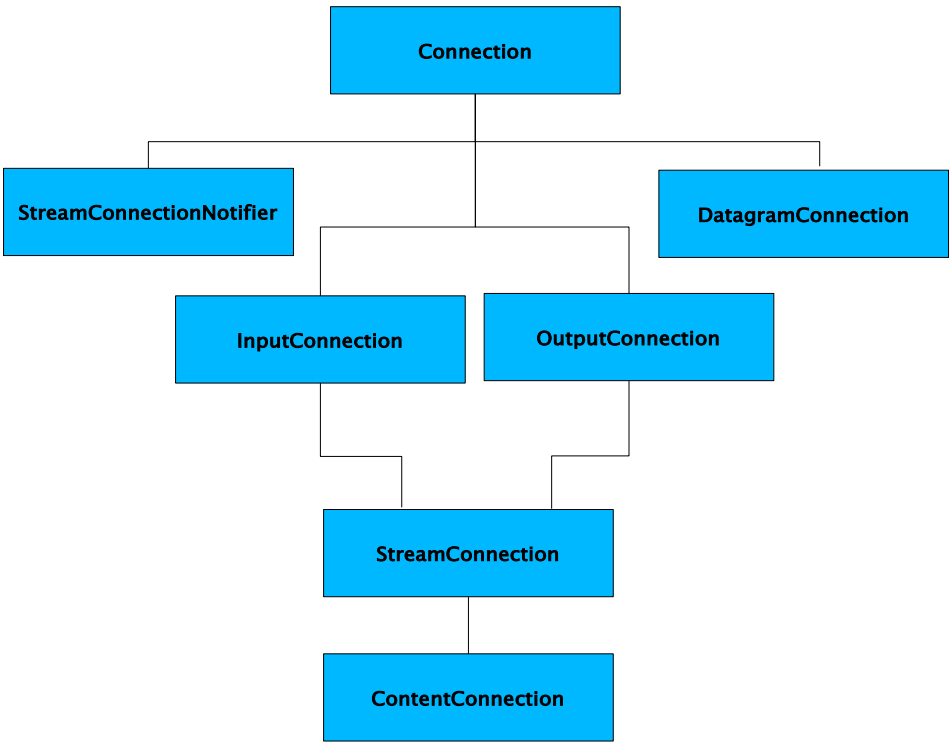

Figura 4: A Hierarquia de Conexão GCF

5. CDC

A Configuração de Dispositivo Conectada (CDC - *Connected Device Configuration*) é um super-conjunto da CLDC. Ela provê um ambiente de execução Java mais amplo que o da CLDC e é um ambiente mais próximo do da JSE.

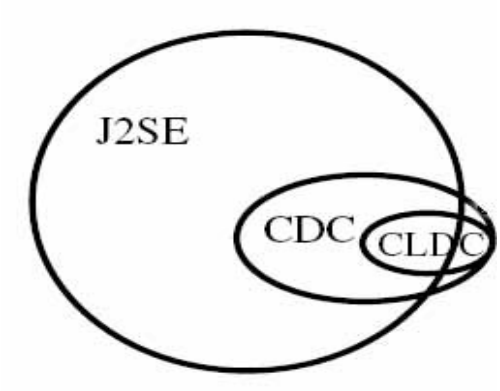

Figura 5: Visão da CDC

A Máquina Virtual Java CDC (ou CVM) é uma Máquina Virtual Java completa. A CDC contém todas as APIs da CLDC. Ela provê um subconjunto maior das classes da JSE.

Como a CLDC, a CDC não define nenhuma classe de interface com o usuário. As bibliotecas de interface com o usuário são definidas pelos perfis desta configuração.

As classes incluídas na CDC vêm dos seguintes pacotes:

- java.io
- java.lang
- java.lang.ref
- java.lang.math
- java.net
- java.security
- java.security.cert
- java.text
- java.util
- java.util.jar
- java.util.zip

CDC também inclui o *Framework* de Conexão Genérica. Ela requer tipos de conexão adicionais como suporte para arquivo e datagrama.

6. JWTI

A Tecnologia Java Para a Indústria Sem Fio (JWTI - *Java Technology for the Wireless Industry*) especifica um conjunto de serviços e especificações padrão. De acordo com a especificação JWTI, seu principal objetivo é "minimizar a fragmentação de APIs no mercado de telefones celulares, e entregar uma especificação clara e previsível para os fabricantes de dispositivos, operadores e desenvolvedores de aplicação".

Por atenderem à JWTI, as aplicações rodarão em um conjunto maior de dispositivos. Os fabricantes de dispositivos irão se beneficiar também porque um conjunto maior de aplicações estará disponíveis para seus dispositivos.

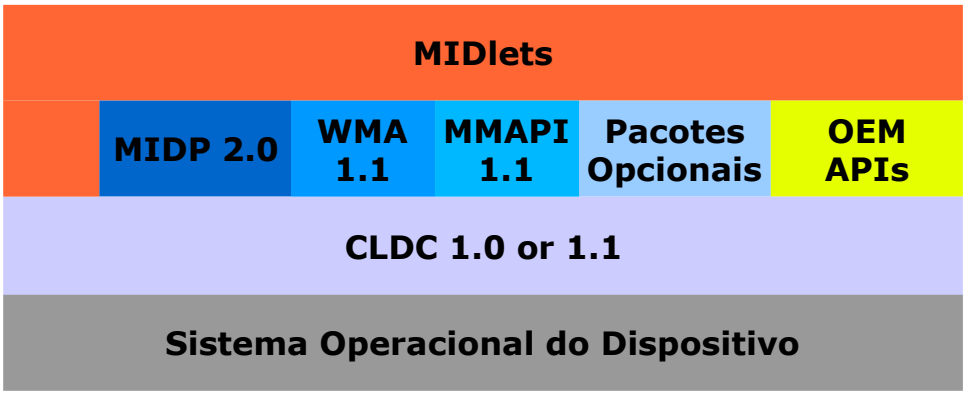

Figura 6: Componentes JWTI

7. MIDP

O Perfil de Dispositivo de Informação Móvel (MIDP - Mobile Information Device Profile) é contruído sobre a CLDC. Não se deve escrever aplicações móveis úteis apenas usando as APIs CLDC. É na MIDP que as APIs de interface com o usuário são definidas.

A especificação MIDP, assim como a CLDC e outras APIs, foi definida através do *Java Community Process* (JCP). Foi envolvido um grupo de profissionais de mais de 50 empresas, composta de fabricantes de dispositivos móveis, operadoras e desenvolvedores de software. A MIDP está continuamente evoluindo, com futuras versões passando pelo mesmo rigor do processo do JCP. Versões futuras do MIDP terão compatibilidade com as versões anteriores, como no caso do MIDP1 e MIDP 2.0.

A especificação MIDP define que um dispositivo MID deve ter as seguintes características, no mínimo:

- Visor:
  - Tamanho da Tela: 96x54
  - Profundidade do Visor: 1-bit
  - *Pixel aspect ratio*: aproximadamente 1:1

- Entrada:
  - Um ou mais dos seguintes mecanismos de entrada: teclado de uma mão, teclado de duas mãos ou tela de toque

- Memória:
  - 256 kilobytes de memória não volátil para a implementação MIDP, mais o que for requerido pela CLDC
  - 8 kilobytes de memória não volátil para os dados persistentes criados pela aplicação
  - 128 kilobytes de memória volátil para o ambiente Java (ex. *Java heap*)

- Rede:
  - Sem fio, duas vias, possivelmente intermitente, com largura de banda ilimitada

- Som:
  - A habilidade de tocar sons, via hardware dedicado ou via software

MIDP define o modelo de aplicação, a API de interface com o usuário, o armazenamento persistente e a rede, API de mídia e jogos, políticas de segurança, entrega da aplicação e provisionamento *over-the-air*.

8. MIDlet

Uma aplicação MIDP é chamada de *MIDlet.* O software de gerenciamento da aplicação (AMS - *Application Management Software*) do dispositivo interage diretamente com o *MIDlet* com os métodos de criar, iniciar, pausar e destruir o *MIDlet.*

O *MIDlet* é parte do pacote *javax.microedition.midlet*. Necessita estender a classe *MIDlet*. E pode requisitar parâmetros do AMS conforme definido no descritor da aplicação (JAD – *Java Application Descriptor*).

Um *MIDlet* não utiliza o método *public static void main(String[] args)*. Caso possua, este não será reconhecido pelo AMS como o ponto de início do programa.

8.1. Ciclo de Vida do MIDlet

A vida de um *MIDlet* começa quando ele é instanciado pelo AMS. Ele inicialmente entra no estado **pausado** após ser criado com comando *new*. O AMS chama o construtor público sem argumento do *MIDlet*. Se uma exceção ocorrer no construtor, o *MIDlet* é colocado no estado **destruído** e é descartado imediatamente.

O *MIDlet* entra no estado **ativo** depois de se chamar o método *startApp*() pelo AMS.

O *MIDlet* entra no estado **destruído** quando o AMS chama o método *destroyApp*(). Este estado também é atingido quando o método *notifyDestroyed*() retorna com sucesso para a aplicação. Observe que o *MIDlet* entra no estado **destruído** somente uma vez no seu tempo de vida.

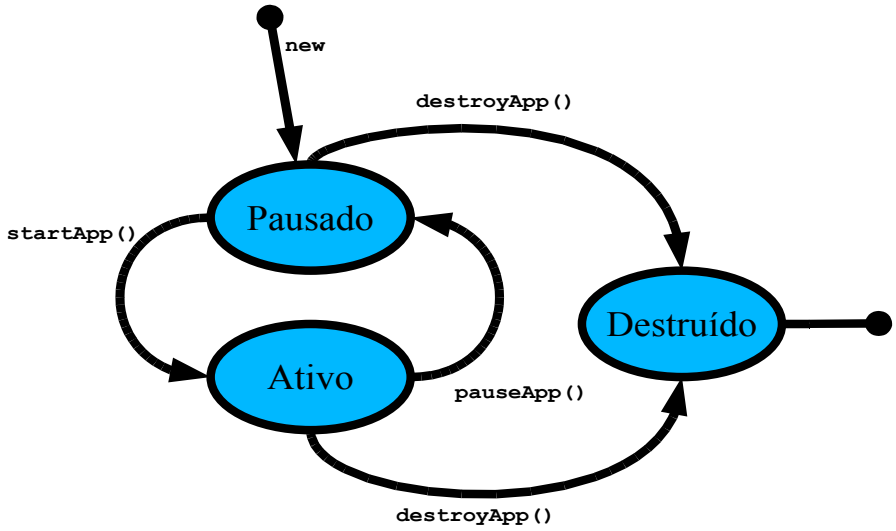

Figura 7: Ciclo de Vida do MIDlet

8.2 MIDlet Suites

As aplicações de MIDP são empacotadas e entregues aos dispositivos como MIDlet suites. Um MIDlet suite consiste em um Arquivo Java (JAR) e, opcionalmente, um descritor de aplicação Java (JAD).

Um arquivo JAD é um arquivo texto contendo um conjunto de atributos, alguns dos quais são requeridos.

9. Exercícios

9.1. Quais são as vantagens do uso de Java como plataforma de desenvolvimento e execução para os dispositivos móveis?

- aplicações altamente portáteis
- interfaces ricas, bem definidas para o dispositivo
- espaço de memória baixa (KVM)
- ambiente execução seguro
- aplicações dinâmicas (podem carregar aplicações para um dispositivo)

9.2. O que o motivaria para escrever para programas para os dispositivos móveis?

- o desafio de escrever aplicações otimizadas
- novos conhecimentos
- fator diversão

Parceiros que tornaram JEDI™ possível

Instituto CTS
Patrocinador do DFJUG.

Sun Microsystems
Fornecimento de servidor de dados para o armazenamento dos vídeo-aulas.

Java Research and Development Center da Universidade das Filipinas
Criador da Iniciativa JEDI™.

DFJUG
Detentor dos direitos do JEDI™ nos países de língua portuguesa.

Banco do Brasil
Disponibilização de seus *telecentros* para abrigar e difundir a Iniciativa JEDI™.

Politec
Suporte e apoio financeiro e logístico a todo o processo.

Borland
Apoio internacional para que possamos alcançar os outros países de língua portuguesa.

Instituto Gaudium/CNBB
Fornecimento da sua infra-estrutura de hardware de seus servidores para que os milhares de alunos possam acessar o material do curso simultaneamente.