

J2ME E O PERFIL MIDP

Por Bruno T Aquino, Roberto Luiz e Ricardo Vinicius Fabri, Abril 14, 2009.

INTRODUÇÃO

Com o passar do tempo os aparelhos celulares deixaram de ser apenas para ligações e agenda para se torna aparelhos realmente bem completos para trabalho, diversão e entretenimento. Os aparelhos passaram a ter várias utilidades como acessar caixa de email e contas bancarias, mas para deixar os aparelhos cada dia mais modernos foram necessárias novas tecnologias de programação e é ai que entra o Java J2ME (Java 2 Micro Edition) uma tecnologia compatível com a maioria do s aparelhos no mercado.

J2ME - Plataforma Java 2, Micro Edition.

Com o passar de o tempo as empresas desidiram deixar a parte de aplicativos dos aparelhos para terceiros, mas isso deixava as empresas responsáveis pelos celulares preocupadas com a exposição de suas tecnologias para outras empresas. Com a chegada do J2ME esse problema acabou, pois bastava as empresas fabricantes dos celulares colocarem uma tecnologia JVM,ou seja, uma maquina virtual Java que carrega e executa aplicativos que as empresas responsável pelos aplicativos poderiam fazer isso sem precisar saber da tecnologia interna de cada aparelho. Assim tanto os fabricantes de celulares integridade de suas tecnologias e os fabricantes dos aplicativos em tempo, pois a maioria dos aplicativos é compatível com a maioria dos aparelhos.

A arquitetura J2ME é basicamente dividida em duas configurações:

- CLDC – Connected Limited Device Configuration;
- CDC – Connected Device Configuration.

O CLDC contém API mínima para execução de aplicativos em dispositivos como celulares. Smartphones, Pager e PDAs. Já CDC está mais lidados a dispositivos com mais

capacidade de memória e processamento como Screen-Phones e alguns PDAs mais poderosos.

Normalmente um CLDC possui geralmente processadores de 16 ou 32 bits e e memória disponível de 128 e 512 KB, para implementações de plataforma Java, já o CDC mais poderoso possui processadores de 32 bits e uma memória de 2MB no mínimo para implementações Java.

Outra parte importante do J2ME são os perfis que consiste em um conjunto de classes que possibilita os desenvolvedores de software implementarem as aplicações de acordo com as características das aplicações dos pequenos dispositivos computacionais. O principal perfil do J2ME é o MIDP (*Mobile Information Device Profile*), que oferece recursos como rede, componentes de interface, armazenamento local, etc.

O MIDP é a definição de uma arquitetura e APIs associadas necessárias para prover um ambiente de desenvolvimento aberto para MIDs (mobile information devices). O MIDP foi feito para rodar em cima do CLDC. Para tanto, um MID deve possuir as seguintes características mínimas de hardware (além daquelas que são requeridas pelo CLDC):

Display:

- Tamanho da tela: 96x54;
- Profundidade: 1 bit;
- Formato do pixel (proporção de aspecto): 1:1;

Input:

- “One handed keyboard” ou
- “Two handed keyboard” ou
- Touch Screen;

Memória:

- 128Kbytes para os componentes MIDP;
- 8Kbytes para dados das aplicações;
- 32Kbytes para o JAVA runtime;

Rede:

- Duplex, sem fio, possivelmente intermitente e com largura de banda limitada.

Os MIDs possuem uma grande variedade de softwares de sistema. Por essa razão, o MIDP estabeleceu alguns requisitos mínimos de sistema:

- Um kernel para controlar o hardware, que possua uma entidade escalonável para rodar a Máquina Virtual Java;
- Um mecanismo para ler e escrever na memória para suportar as APIs;
- Acesso de leitura e escrita à rede sem fio;
- Um mecanismo que provenha um tempo-base utilizado no timestamping nas escritas na “persistent storage”;
- Capacidade de escrever num display bit-mapped;
- Um mecanismo para capturar entrada de um input device;

Como os MIDs possuem uma grande quantidade de potencialidades, o MIDPEG (grupo que elaborou o MIDP) limitou o conjunto de APIs necessárias para apenas aquelas necessárias para alcançar uma grande portabilidade. São as seguintes:

- Aplicação;
- Interface do Usuário;
- Persistent Storage;
- Rede;
- Temporizadores (timers);

As APIs MIDP rodam em cima do CLDC, como foi explicado anteriormente e pode-se visualizar na figura. As classes OEM são classes não definidas pelo MIDP e podem ser utilizadas para funcionalidades específicas para determinado aparelho, o que significa que elas podem ou não ser portáteis para outros MIDs.

Quanto aos aplicativos, um MIDlet é aquele que usa APIs definidas pelo MIDP e CLDC, e são portáteis entre vários aparelhos. Uma aplicação OEM são aquelas que não fazem parte da especificação. Já as nativas são as que estão implementadas diretamente no sistema e não são escritas em Java.

Temporizadores

Aplicações que necessitem agendar tarefas para um tempo futuro pode utilizar as classes Timer e TimerTask, que incluem funções para uma execução ou várias execuções em determinados intervalos. Elas estão em `java.util.Timer` e `java.util.TimerTask`.

Rede

O MIDP herda a conectividade do CLDC e suporta um subconjunto do HTTP, que pode ser implementado com protocolos IP (TCP/IP) e não-IP (WAP e i-mode).

Como há uma grande variedade de redes sem fio, fica por responsabilidade do aparelho e da própria rede sem fio de providenciar o serviço de aplicação. Pode ser necessário um gateway para ligar a rede sem fio à internet. A aplicação cliente não deve precisar saber se há alguma rede não-IP sendo usada ou qualquer outra característica de outras redes. No entanto, essa possibilidade existe, o que faria o cliente e o servidor tirar vantagem deste conhecimento otimizando suas transmissões.

A interface `HttpConnection` possui funcionalidades que permitem a realização de funções específicas do HTTP. Qualquer aparelho que implemente MIDP deve suportar o HTTP 1.1, requisições HEAD, GET, POST e forms.

Persistent Storage

O MIDP disponibiliza um mecanismo para que as MIDlets possam guardar dados e lê-los mais adiante. É a chamada Record Management System (RMS).

Record Stores:

É uma coleção de registros que permanece o mesmo durante múltiplas chamadas do MIDlet. A plataforma é responsável por manter a integridade desses registros, mesmo após reboots ou trocas de baterias.

Records:

São vetores de bytes. Utilizados para armazenagem de diferentes tipos de dados. Eles são unicamente identificados pelo seu `recordId`, um valor inteiro.

Aplicações

O MIDP define um modelo de aplicação que permite que os recursos limitados dos MIDs sejam compartilhados por várias aplicações, as MIDlets. Este compartilhamento é viável mesmo com os limitados recursos e framework de segurança do MID pois eles são

obrigados a compartilhar classes e estão sujeitos a um conjunto de políticas e controles que permitem isso.

Os elementos de uma MIDlet suite, dos quais se espera que implementem as funções necessárias pelos usuários para instalar, selecionar, rodar e remover midlets, são (seu conjunto forma o software de gerenciamento de aplicação):

Ambiente de Execução: é compartilhado por todas as MIDlets que estão na mesma MIDlet suite, e qualquer MIDlet pode interagir com outra que esteja no mesmo pacote.

Empacotamento do MIDlet suite: uma ou mais MIDlets podem ser empacotadas num único arquivo JAR, que contém as classes compartilhadas e os arquivos de recursos utilizados pelas MIDlets, além de um manifesto descrevendo seu conteúdo. Existem vários atributos pré-definidos que permitem identificação de uma MIDlet, como nome, versão, tamanho de dados, descrição, etc.

Descritor de Aplicação: é utilizado para gerenciar a MIDlet e é usada pela própria MIDlet para atributos de configuração específica. O descritor permite que seja verificado que a MIDlet é adequada ao aparelho antes de se carregar todo o arquivo JAR da MIDlet suite. Ele também permite que parâmetros sejam passados para as MIDlets sem modificar os arquivos JAR.

Ciclo de Vida da Aplicação: uma MIDlet não deve possuir um método public void static main (). O software de gerenciamento de aplicação deve suprir a classe inicial necessária pelo CLDC para iniciar a MIDlet. Quando uma MIDlet é instalada, ela é mantida no aparelho e fica pronta para uso. Quando é rodada, uma instância é criada através de seu construtor público sem argumentos, e seus métodos são chamados para mudar passar pelos estados da MIDlet. Quando ela é terminada, é destruída, e os recursos utilizados por ela podem ser recuperados, incluindo os objetos criados e suas classes.

O software de gerenciamento de aplicação disponibiliza um ambiente no qual a MIDlet é instalada, iniciada, parada e desinstalada. Também é responsável por manusear os erros que podem ocorrer durante alguma destas etapas.

O compartilhamento de dados e outras informações entre MIDlets é controlado pelas APIs individuais e suas implementações. Assim, por exemplo, os métodos da API de um sistema de gerenciamento de registros devem ser especificados para manusear com dados que podem ser compartilhados com outras MIDlets.

Referencias Bibliográficas:

http://pt.wikipedia.org/wiki/Java_ME

<http://imasters.uol.com.br/artigo/1539/java/j2me - java para os portateis/>

<http://grenoble.ime.usp.br/movel/j2me.pdf>

<http://www.htmlstaff.org/ver.php?id=56>