Java ME

Aplicaciones Móviles
Curso de Adaptación
Grado en Ingeniería de Sistemas Audiovisuales

Celeste Campo - Carlos García Rubio
celeste, cgr@it.uc3m.es

Índice

- Introducción
- Plataforma Java ME:
  - Java ME dispositivos menos limitados.
  - Java ME dispositivos limitados.
- Programación Java ME dispositivos limitados:
  - Conceptos generales de MIDlets.
  - Desarrollo de MIDlets:
    - MIDlet: ¡Hola mundo!
  - Verificación de aplicaciones Java Verified™
- Referencias

Introducción

- Java es un entorno de programación orientado a objetos desarrollado por Sun:
  - *"write once, run anywhere"*
- Cuatro ediciones:
  - Java EE (Enterprise Edition):
    - Construir aplicaciones distribuidas para entornos empresariales.
    - Centrado en el desarrollo del lado del servidor.
    - Aplicaciones web.
  - Java SE (Standard Edition):
    - Construir aplicaciones en entornos PC.
  - **Java ME (Micro Edition)**:
    - Subconjunto de Java SE para dispositivos embebidos (móviles, PDAs, TV set-top box).
  - Java Card:
    - Construir aplicaciones para tarjetas inteligentes.
    - También para dispositivos móviles a través de las tarjetas (U)SIM.

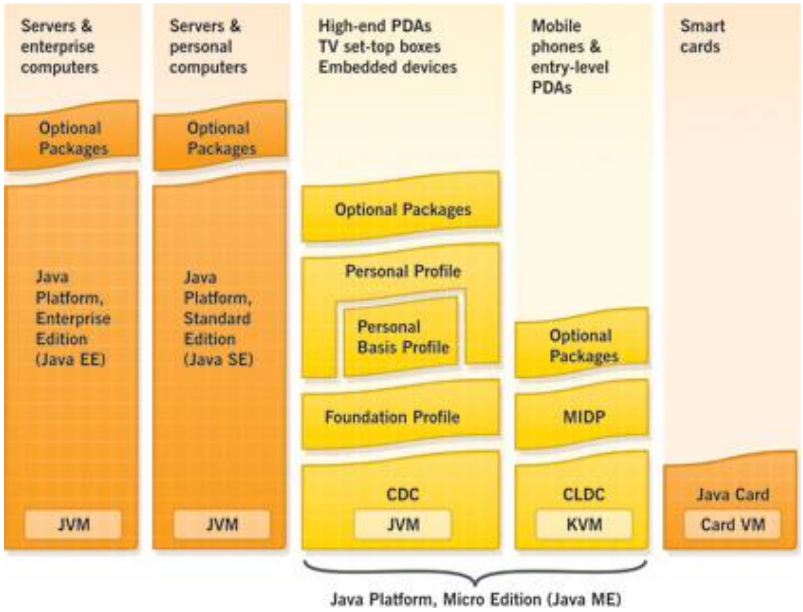
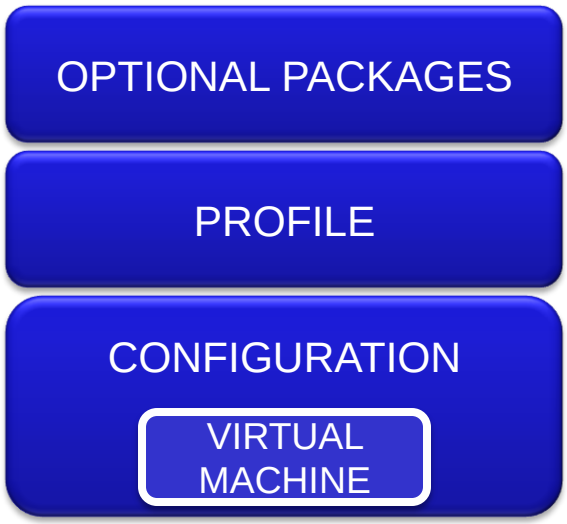


Imagen obtenida de http://netbeans.org/community/magazine/html/03/mobpack/

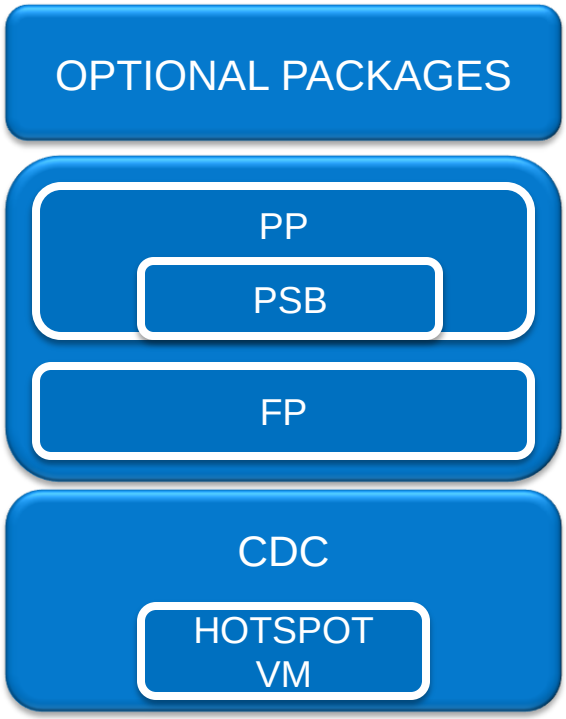
Plataforma Java ME

- Java ME está dirigido a un amplio rango de dispositivos.
- Para conseguir flexibilidad y adaptación se estructura en tres niveles:
  - **Configuración**.
  - **Perfil**.
  - **Paquetes opcionales.**
- Estandarización realizada vía la iniciativa *Java Community Process* (JCP) que genera los correspondientes *Java Specification Report* (JSR).
- Un mismo dispositivo puede soportar varios perfiles con varios paquetes opcionales.



Java ME: Dispositivos menos limitados

- Características:
  - Interfaces de usuario más complejas.
  - Memoria en el rango de 2 – 16 Mb para Java.
  - Conectividad (no sólo inalámbrica)
  - Procesadores de 16 o 32 bits.
  - Ejemplos: Internet TV, TV set-top boxes y PDAs.
- Plataforma Java ME:
  - Configuración:
    - **CDC (Connected Device Configuration)**.
    - **CDC Hotspot VM**.
  - Perfiles:
    - **FP (Foundation Profile)**, **PSB (Personal Basic Profile)** y **PP (Personal Profile).**
  - Paquetes opcionales.



Java ME: Dispositivos menos limitados

- CDC:
  - Versiones 1.0 (JSR 36) y 1.1 (JSR 218).
  - Soporte completo al lenguaje Java y la especificación de JVM.
  - Compatible con CLDC:
    - Incompatibilidades aparecen a nivel perfil.

PAQUETES	DESCRIPCIÓN
`java.io`	Clases e interfaces estándar de E/S.
`java.lang`	Clases e interfaces básicas del lenguaje
`java.math`	Clases soporte matemático
`java.net`	Clases e interfaces de red (TCP/IP)
`java.security`	Clases e interfaces de seguridad y gestión de certificados
`java.text`	Clases e interfaces para gestionar textos, número, fechas, etc
`java.util`	Clases, interfaces y utilidades estándar.
`javax.microedition.io`	Clases e interfaces de conexión genérica dispositivos limitados

Java ME: Dispositivos menos limitados

- FP:
  - Versiones 1.1.2 (JSR 219) y 1.0 (JSR 46):
    - La versión 1.1.2 añade mayor soporte a APIs de seguridad y de comunicaciones.
  - Incluye APIs de comunicación, seguridad, compresión y utilidades como temporizadores y gestión de eventos.
  - No incluye ningún soporte de interfaz gráfico.
  - Ejemplos: impresoras, routers, pasarelas residenciales,…
- PBP:
  - Versiones 1.1.2 (JSR 217) y 1.0 (JSR 129):
    - La versión 1.1.2 añade más clases de interfaz gráfico.
  - Necesita el FP, por lo tanto, engloba las APIs de FP.
  - Incluye APIs para el soporte de interfaces gráficas básicas (basado en AWT), soporte a JavaBeans y el modelo de programación de Xlets.
    - El soporte de GUI está derivado de Java SE 1.4.
  - Ejemplos: TV interactiva, automóviles, videocámaras,…

Java ME: Dispositivos menos limitados

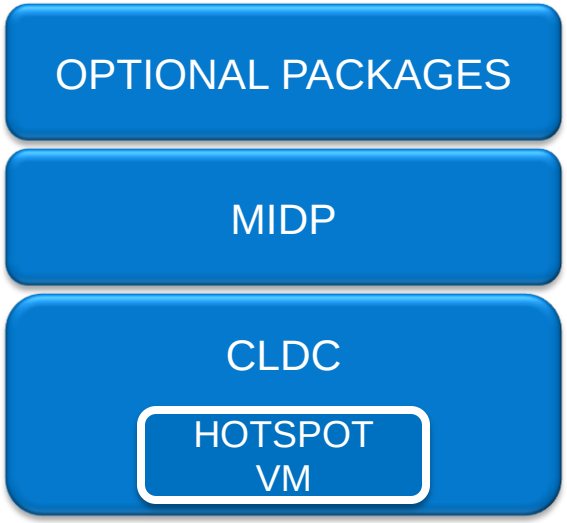
- PP:
  - Versiones 1.1.2 (JSR 216) y 1.0 (JSR 62):
    - La versión 1.1.2 añade más clases de interfaz gráfico.
  - Necesita el PBP, por lo tanto, engloba las APIs de PBP y FP.
  - Incluye soporte completo a AWT y soporte a applets.
  - Ejemplos: PDAs de altas prestaciones, navegadores web embebidos,…

Java ME: Dispositivos menos limitados

- Paquetes opcionales específicos de CDC:
  - *RMI (JSR 66)*:
    - Sobre FP.
    - Invocación de métodos de forma remota basada en Java.
    - Subconjunto de Java SE (`java.rmi`).
  - *JDBC (JSR 169)*:
    - Soporte para conexión a bases de datos relacionales.
    - Basado en el API JDBC 3.0 de Java SE (subconjunto de `java.sql`, `javax.sql`).
  - *AGUI (JSR 209)*:
    - Sobre PBP (y por lo tanto, FP).
    - Soporte para interfaces swing, extensiones de Java 2D, manipulación de imágenes y gestión de “look and feel”.
- Otros con soporte también para CLDC.

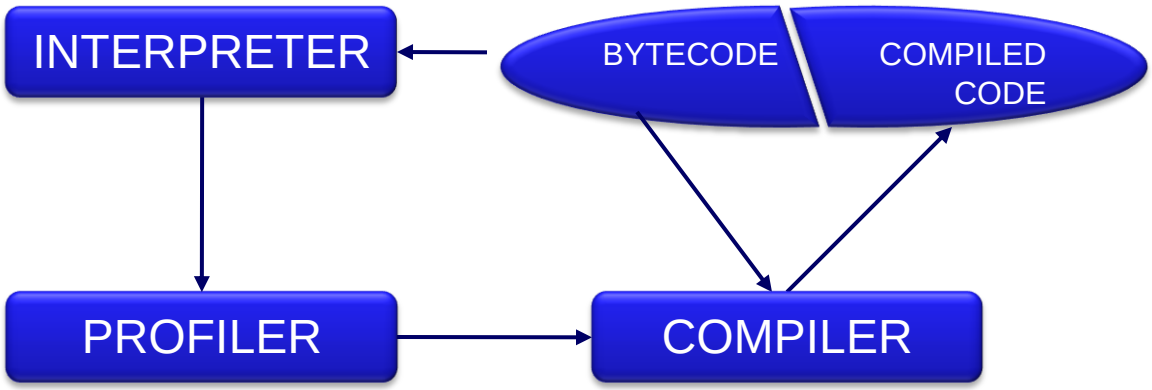
Java ME: Dispositivos limitados

- Características:
  - Interfaces de usuarios sencillas.
  - Memoria en el rango de 160 – 512 kb para Java.
  - Comunicaciones inalámbricas.
  - Procesadores de 16 o 32 bits.
  - Funcionamiento con baterías.
  - Ejemplos: móviles y PDAs.
- Plataforma Java ME:
  - Configuración:
    - **CLDC (Connected Limited Device Configuration)**
    - **CLDC Hotspot VM.**
  - Perfil:
    - **MIDP (Mobile Information Device Profile).**
  - Paquetes opciones.



Java ME: Dispositivos limitados

- CLDC HotSpot VM:
  - Máquina virtual mejorada (sustituye a las anteriores KVM y CVM).
  - Basada en el concepto de compilación dinámica de “hotspot” a código nativo:
    - Un “hotspot” son trozos de código que se repiten frecuentemente o que tienen requisitos temporales críticos.



Java ME: Dispositivos limitados

- CLDC:
  - Versiones 1.0 (JSR 30) y 1.1 (JSR 139)
    - La versión 1.1 es compatible con la 1.0.
    - La 1.1 añade mejoras de prestaciones, soporte matemático de coma flotante y a referencias débiles.
  - Incluye el entorno de ejecución y un conjunto de clases básicas.
  - No incluye ningún API de gestión de interfaces gráficas, ni de gestión del ciclo de vida de las aplicaciones, ni implementa modelos de E/S:
    - Fuertemente ligado a MIDP.

PAQUETES	DESCRIPCIÓN
`java.io`	Clases estándar de E/S. Subconjunto de Java SE.
`java.lang`	Clases e interfaces de la VM. Subconjunto de Java SE
`java.util`	Clases, interfaces y utilidades estándar. Subconjunto de Java SE
`javax.microedition.io`	Clases e interfaces de conexión genérica CLDC

Java ME: Dispositivos limitados

- MIDP:
  - Versiones 1.0 (JSR 37), 2.0 (JSR 118) y 3.0 (JSR 271):
    - 1.0: soporte de interfaz gráfico, almacenamiento y conectividad.
    - 2.0: compatible con 1.0, añade mejoras de interfaz gráfico, multimedia, juegos y conectividad.
    - 3.0: compatible con 2.0, nuevo concepto de LIBlets (LIBrary), mejoras de interfaz gráfica, seguridad, almacenamiento RMS, categorización de aplicaciones, concurrencia y comunicación entre MIDlets, etc…
      - Estándar desde Diciembre de 2009.
      - Requisitos mayores que las versiones anteriores.
  - Complementa CLDC añadiendo gestión del ciclo de vida de las aplicaciones (MIDlets), interfaz gráfico, conectividad, almacenamiento persistente, multimedia (sonidos) y seguridad.

Java ME: Dispositivos limitados

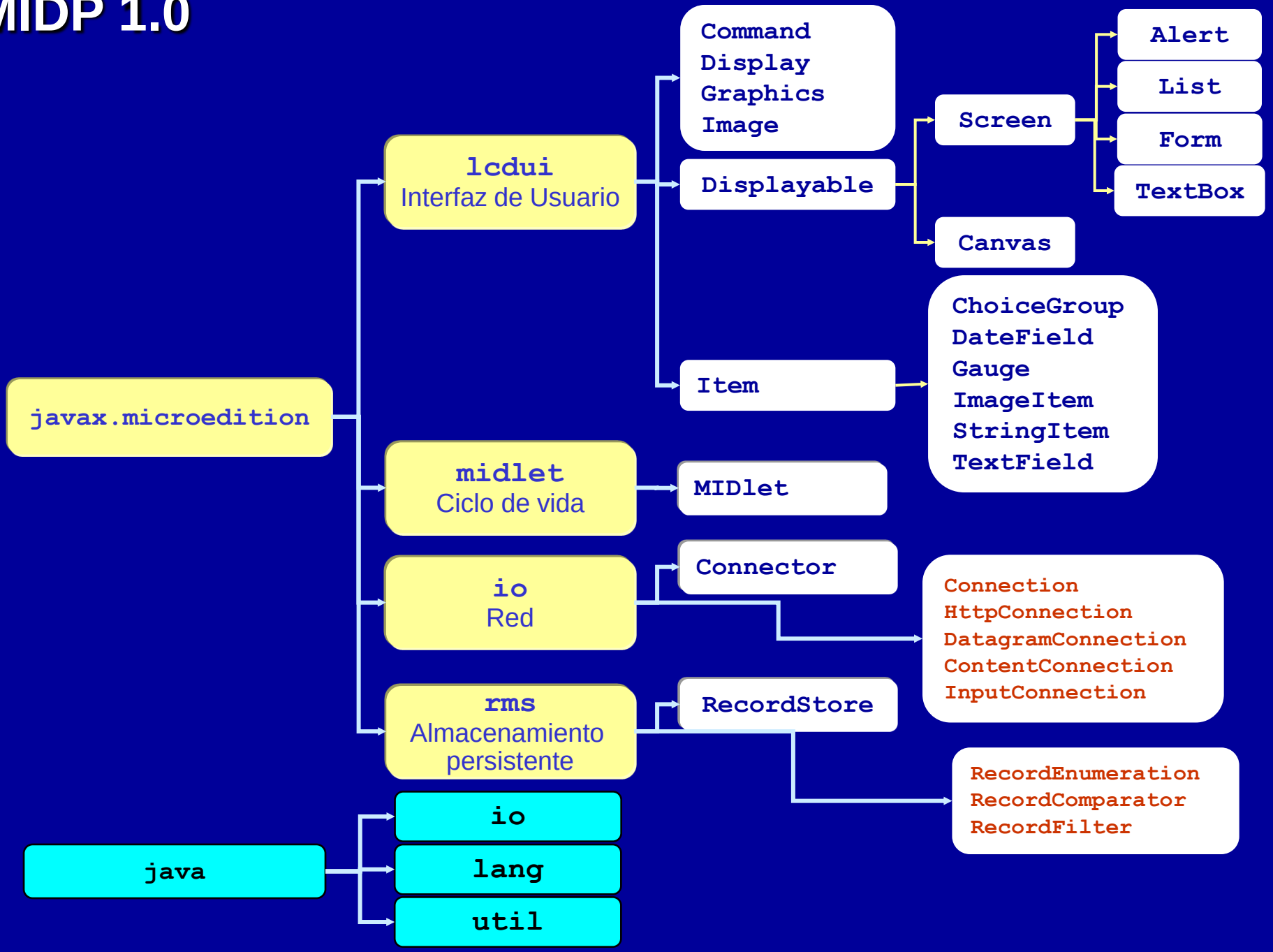
MIDP 1.0

PAQUETES	DESCRIPCIÓN
`javax.microedition.midlet`	Ciclo de vida de las aplicaciones (MIDlets)
`javax.microedition.lcdui`	Clases e interfaces de interfaz gráfica
`javax.microedition.rms`	Clases, interfaces de almacenamiento persistente
`javax.microedition.io`	Clases e interfaces de conexión.

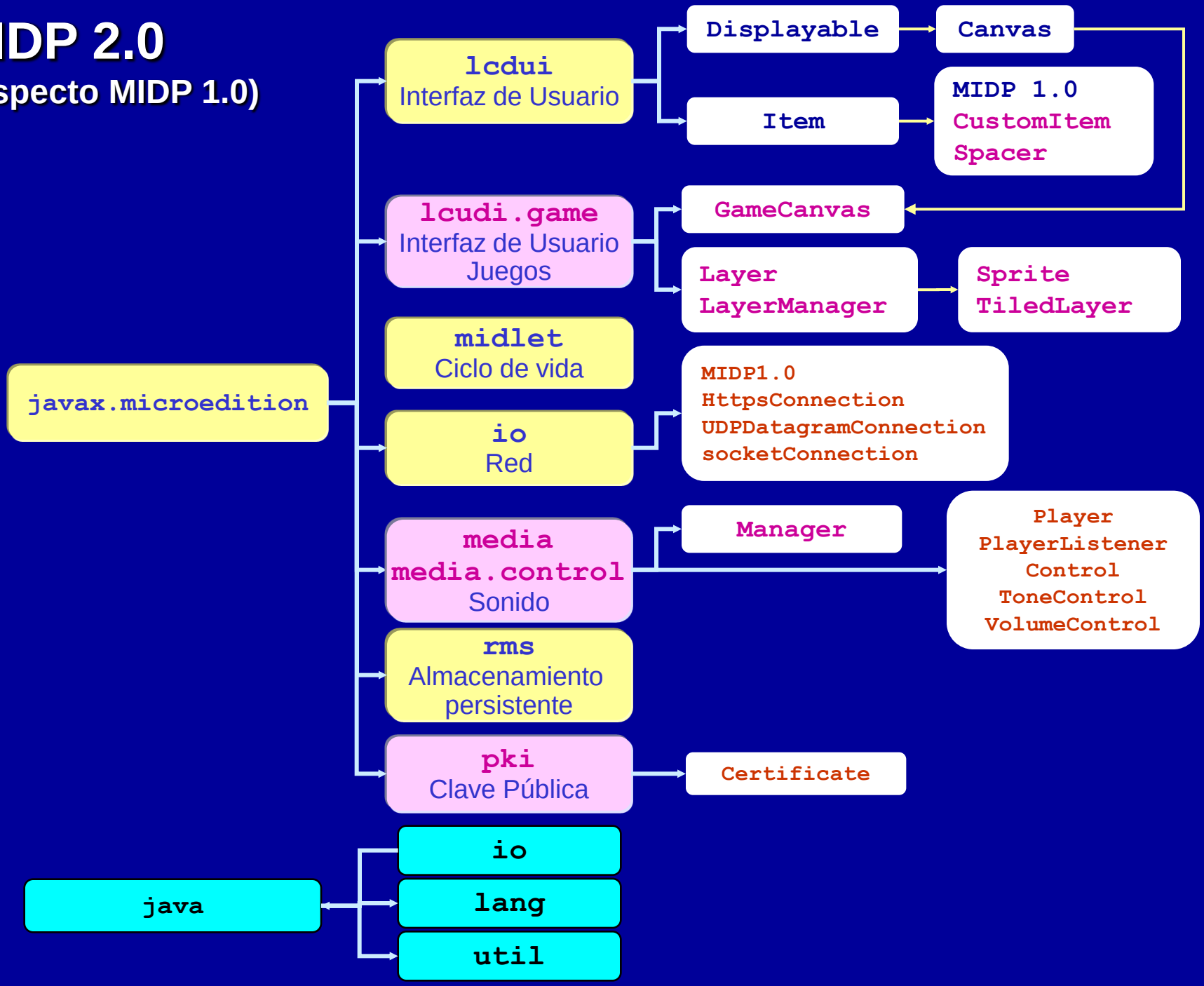
MIDP 2.0

PAQUETES	DESCRIPCIÓN
`javax.microedition.lcdui.game`	Clases e interfaces de interfaz gráfica para juegos
`javax.microedition.pki`	Clases e interfaces de seguridad basada en clave pública
`javax.microedition.media` `Javax.microedition.media.control`	Clases, interfaces para reproducción de sonido

MIDP 1.0


javax.microedition
lcdui
Interfaz de Usuario
Command
Display
Graphics
Image
Displayable
Screen
Alert
List
Form
TextBox
Canvas
Item
ChoiceGroup
DateField
Gauge
ImageItem
StringItem
TextField
midlet
Ciclo de vida
MIDlet
io
Red
Connector
Connection
HttpConnection
DatagramConnection
ContentConnection
InputConnection
rms
Almacenamiento
persistente
RecordStore
RecordEnumeration
RecordComparator
RecordFilter
java
io
lang
util

MIDP 2.0
(respecto MIDP 1.0)



Java ME: Dispositivos limitados

- Paquetes opcionales de **interfaz gráfica**:
  - *MMAPI: Mobile Media API (JSR 135)*:
    - Soporte de audio, video y objetos multimedia (reproducción y grabación).
    - Es importante chequear primero las capacidades multimedia del propio dispositivo mediante las propiedades del sistema.
    - Válida para CLDC y CDC, sobre varios perfiles.
  - *Advanced Media Supplements (JSR 234)*:
    - Complementaria a la MMAPI.
    - Acceso a controles específicos de la cámara (flash, zoom, brillo, contraste, etc.), acceso a emisoras de radio, capacidades multimedia avanzadas (ecualizador, efectos de audio, reverberación, audio 3D, etc.) y selección del dispositivo de salida (altavoz/auricular).
  - *Mobile 3D Graphics (JSR 184)*:
    - Soporte para gráficos interactivos 3D en dispositivos limitados.
    - No requiere hardware específico 3D en el dispositivo.
    - Válida para CLDC/MIDP, aunque se puede utilizar en otros perfiles.

Java ME: Dispositivos limitados

- Paquetes opcionales de **interfaz gráfica**:
  - *Scalable 2D Vector Graphics (JSR 226)*:
    - Soporte para visualización 2D a partir de formatos W3C *Scalable Vector Graphics* (SVG): visualización de mapas, iconos escalables, etc.
    - No requiere hardware específico 2D en el dispositivo.
    - Válida para CLDC/MIDP, aunque se puede utilizar en otros perfiles.
  - *Java Binding for Open GL ES (JSR 239)*:
    - Acceso Java al soporte nativo de gráficos 3D a través de Open GL ES.
    - Permite acceso a hardware específico, soportado en algunos dispositivos móviles para el desarrollo de juegos.
    - Misma funcionalidad que Mobile 3D Graphics (JSR 184) pero se facilita la “traducción” a Java ME de (numerosas) aplicaciones que utiliza Open GL ES.
    - Válida para CLDC/MIDP, aunque se puede utilizar en otros perfiles.

Java ME: Dispositivos limitados

- Paquetes opcionales de **interfaz gráfica**:
  - *Lightweight UI Toolkit* (LWUIT):
    - Mejorar el soporte de UI en Java ME.
    - Se soporta a partir de CLDC 1.1 y MIDP 2.0.
    - Fácil de utilizar si se conoce AWT o Swing de Java:
      - *Layouts*.
      - Eventos y Listeners.
      - Incluir y modificar temas en los interfaces.
      - Efectos de transiciones entre pantallas.

Java ME: Dispositivos limitados

- Paquetes opcionales de **comunicaciones**:
  - *WMA: Wireless Messaging API (JSR 120, 205)*:
    - Versión 1.0 (JSR 120):
      - Soporte a protocolos basados en mensajería, como SMS.
    - Versión 2.0 (JSR 205):
      - Soporte al envío y recepción de mensajes multimedia MMS.
    - Válido para CLDC/MIDP.
  - *SIP (JSR 180)*:
    - Soporte a SIP (establecimiento y gestión de sesiones multimedia), utilizado para el desarrollo de aplicaciones de mensajería, presencia y juegos.
    - Válida para CLDC/MIDP, aunque se puede utilizar en otros perfiles.
  - *Bluetooth (JSR 82)*:
    - Soporte a conexiones Bluetooth:
      - Al menos debe soportar RFCOMM, OBEX y los protocolos de descubrimiento de servicios (SDP).
    - Válido para CLDC, normalmente con MIDP pero no es necesario.

Java ME: Dispositivos limitados

- Paquetes opcionales de **información personal**:
  - *PDA Optional Package for Java ME (JSR 75)*:
    - Define dos paquetes independientes:
      - Personal Information Management (PIM), de acceso a datos de agenda, calendarios, guía, listas de tareas, etc. del usuario.
      - File Connection (FC), permite acceso al sistema de ficheros nativo del dispositivo.
    - Válido para CLDC/MIDP.
  - *Location (JSR 179)*:
    - Soporte al desarrollo de aplicaciones basadas en localización.
    - No está ligada a la obtención de localización a través de una determinada tecnología (ej. GPS), es un API genérica.
    - Válida para CLDC y CDC, sobre varios perfiles.

Java ME: Dispositivos limitados

- Paquetes opcionales de **comunicación con aplicaciones**:
  - *CHAPI: Content Handler API (JSR 211)*:
    - Clases y modelo para gestionar acciones asociadas a URI basándose en tipos MIME.
      - Dependiendo del tipo de contenido MIME se lanza una determinada aplicación (Java ME o no) identificada por una URI.
    - Válido para CLDC/MIDP y CDC/PP.
  - *WSA: Web Services API (JSR 172)*:
    - Soporte al desarrollo de clientes de servicios web basados en tecnologías estándar del W3C y Oasis (procesado XML, WSDL, XML-RPC).
    - Válido para CLDC y CDC.

Java ME: Dispositivos limitados

- Paquetes opcionales de **seguridad y comercio electrónico**:
  - *Payment (JSR 229)*:
    - Soporte para realizar pagos electrónicos a través del móvil.
    - Válido para CLDC y CDC.
  - *SATSA: Security and Trust Services API (JSR 177)*:
    - Soporte para proporcionar servicios de seguridad:
      - Almacenamiento seguro de claves, certificados personales, credenciales de servicio e información personal.
      - Ejecución segura de aplicaciones (algoritmos criptográficos, integridad y confidencialidad de los datos).
      - Autenticación e identificación para el acceso a aplicaciones Java ME.
    - Válido para CLDC y CDC.

Java ME: Dispositivos limitados

- Otros paquetes opcionales:
  - *Contactless Communications (JSR 257)*:
    - Soporte a lectura/escritura de RFID, comunicación de corto alcance basada en NFC y lectura de códigos de barras.
    - Válida para CLDC y CDC, sobre varios perfiles.
  - *Mobile Sensor API (JSR 256)*:
    - Soporte unificado para la gestión y captura de datos obtenidos por los sensores del dispositivo móvil (acelerómetro, temperatura, luz, proximidad, etc.)
    - Válida para CLDC y CDC, sobre varios perfiles.

Java ME: Dispositivos limitados

- Conjuntos de paquetes opcionales para la industria de telefonía móvil:
  - *JTWI: Java Technology for Wireless Industry (JSR 185)*:
    - Primera estandarización, incluía CLDC 1.0 o 1.1 / MIDP 2.0 / WMA 1.1 y MMAPI.
  - *Mobile Service Architecture (JSR 248)*:
    - Evolución de JTWI, en la que se define un conjunto de APIs básico (subset) y un conjunto más completo.
    - MSA subset: CLDC 1.1 / MIDP 2.0 / JSR 75 / JSR 82 / JSR 135 / JSR 205 y JSR 226.
    - MSA: MSA subset / JSR 172 / JSR 177 / JSR 179 / JSR 180 / JSR 211 y JSR 229.

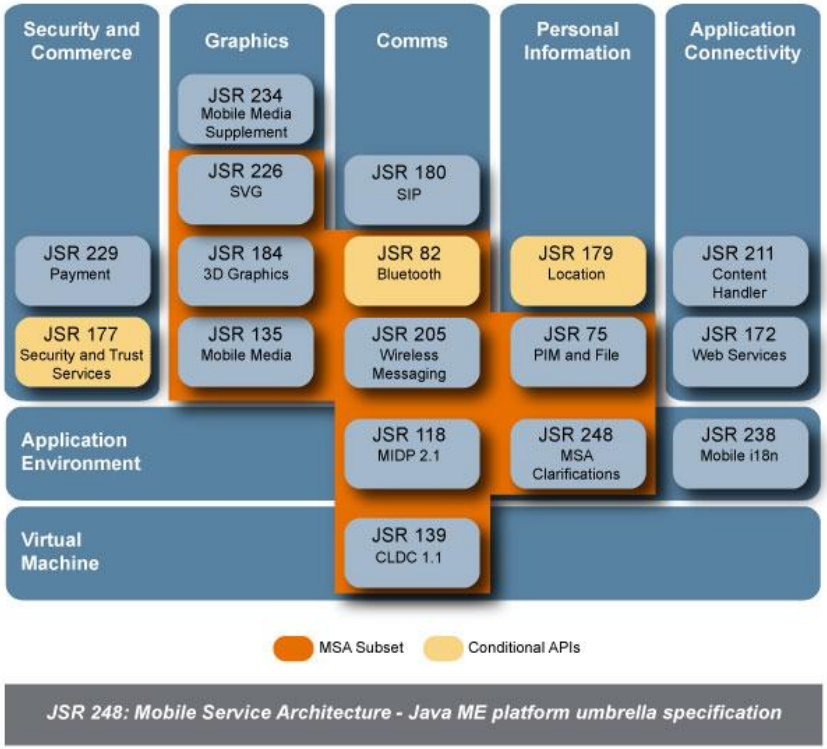


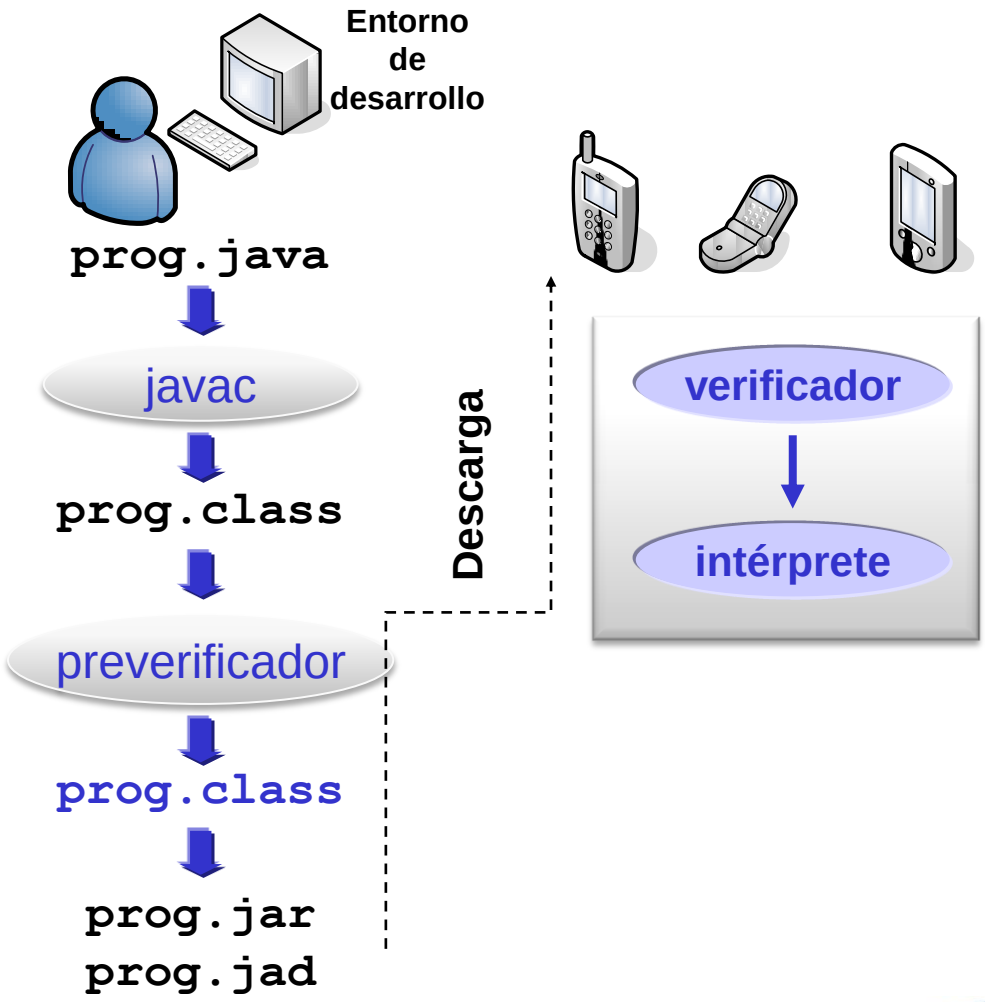
Imagen obtenida de http://crmondemand.oracle.com/technetwork/java/javame/tech/msa-139431.html

Programación Java ME: MIDlets

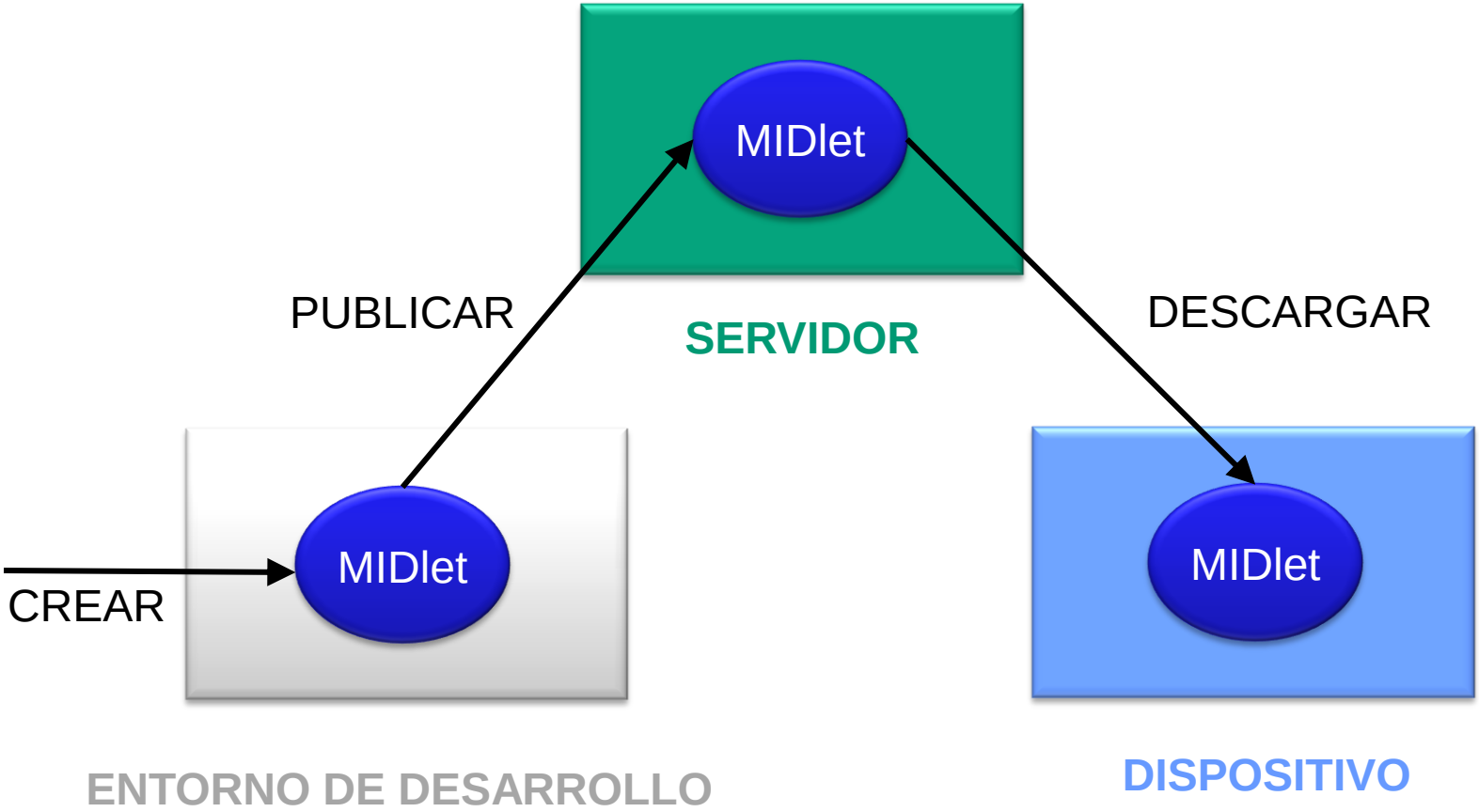
- ¿Qué es un MIDlet?
  - Aplicación MIDP.
- El desarrollo de un MIDlet consta de las siguientes etapas:
  - Creación.
  - Publicación.
  - Descarga e instalación.
  - Ejecución.
  - Actualización (gestión de versiones) y borrado.

MIDlet: Creación

- Escribir el código y compilar.
- Preverificar el código.
- Empaquetar en un JAR y crear el descriptor (fichero JAD).
- Ejecutar en emulador.



MIDlet: Publicación



MIDlet: Descarga e instalación

- Descarga:
  - Gestionado por el *Application Management System (AMS)*.
  - El dispositivo obtiene el MIDlet de alguna fuente, a través:
    - Red inalámbrica, puerto USB, IrDA, Bluetooth, etc..
  - Se establece una negociación sobre capacidades del dispositivo según los requisitos del MIDlet, coste...:
    - El resultado de esta negociación puede provocar que no se realice una descarga efectiva.
  - Se descarga el MIDlet a la memoria del dispositivo:
    - Primero se descarga el descriptor (JAD).
    - Después se descarga el paquetes (JAR).
- Instalación:
  - Gestionado por el *AMS*.
  - Puede comprobar que el MIDlet no vulnera las políticas de seguridad del móvil.
  - Puede transformar (convertir) el MIDlet de formato “público” a un formato específico del dispositivo.
  - Se extrae la aplicación del paquete (JAR) y se deja en disposición de ser ejecutada.

MIDlet: Ejecución

- El usuario selecciona el MIDlet y lo ejecuta.
- En este momento, el MIDlet entra en la VM y se invocan los métodos que gestionan su ciclo de vida:
  - **Paused**: Iniciado y a la espera.
  - **Active**: Tiene recursos ejecutando.
  - **Destroyed**: Ha liberado recursos, destruido hilos y terminado toda su actividad.

MIDlet: Actualización y borrado

- Actualización:
  - Puede publicarse una nueva versión del MIDlet.
  - *AMS* debe gestionar la lista de MIDlets instalados y sus versiones.
    - Puede así actualizar de versiones más antiguas a más recientes del MIDlet.
  - Los atributos del MIDlet, incluida la versión, están:
    - En el descriptor del MIDlet (JAD).
    - En el manifiesto del MIDlet contenido en el JAR.
- Borrado:
  - A través del AMS puede borrarse un MIDlet cuando no va a utilizarse más.
  - Se borra:
    - MIDlet.
    - Todos los registros en memoria permanente escritos por ese MIDlet.

MIDlet: MIDlet Suite

- MIDlet Suite:
  - Conjunto de MIDlets contenidas en un mismo .JAR.
    - El fichero .JAR cumple el formato estándar.
  - Incluye los ficheros de clases y otros recursos asociados al MIDlet, por ejemplo imágenes.
- Normalmente aunque sólo se incluya un MIDlet se trabaja con MIDlet Suite.
- Los MIDlets en un MIDlet Suite comparten recursos:
  - Contenidos en el JAR.
  - RMS creados por los otros MIDlets del mismo MIDlet Suite.

MIDlet: Manifiesto

- El manifiesto (`manifest.mf`) está incluido en el JAR y contiene información sobre los contenidos del fichero JAR.
- Archivo de texto con la estructura: `atributo:valor`

ATRIBUTOS OBLIGATORIOS	ATRIBUTOS OPCIONALES
MIDlet-Name	**MIDlet-Description**
MIDlet-Version	**MIDlet-Icon**
MIDlet-Vendor	**MIDlet-Info-URL**
MIDlet-<n> (name, icon, class)	**MIDlet-Data-Size**
MicroEdition-Profile	**MIDlet-Permissions**
MicroEdition-Configuration	**MIDlet-Permissions-Opt**
	MIDlet-Push-<n>
	MIDlet-Install-Notify
	MIDlet-Delete-Notify
	MIDlet-Delete-Confirm

MIDlet: Descriptor

- Fichero de extensión .JAD y que permite al AMS comprobar si el MIDlet es adecuado para descargarlo.
- Sigue el mismo formato que un manifiesto

ATRIBUTOS OBLIGATORIOS	ATRIBUTOS OPCIONALES
MIDlet-Name	***MIDlet-<n> (name, icon, class)***
MIDlet-Version	***MicroEdition-Profile***
MIDlet-Vendor	***MicroEdition-Configuration***
MIDlet-Jar-URL	**MIDlet-Description**
MIDlet-Jar-Size	**MIDlet-Icon**
	MIDlet-Info-URL
	MIDlet-Data-Size
	MIDlet-Permissions
	MIDlet-Permissions-Opt
	MIDlet-Push-<n>
	MIDlet-Install-Notify
	MIDlet-Delete-Notify
	MIDlet-Delete-Confirm

MIDlet: `javax.microedition.midlet.MIDlet`

- Clase abstracta base para todos los MIDlets:
  - **`protected MIDlet()`**
    - Constructor para crear un MIDlet, sin argumentos.
  - **`protected abstract void startApp() throws MIDletStateChangeException`**
    - Se le llama cuando el MIDlet comienza (paused -> active state)
  - **`protected abstract void pauseApp()`**
    - Se le llama cuando el MIDlet pausa (active -> paused state)
  - **`protected abstract void destroyApp(boolean unconditional)throws MIDletStateChangeException`**
    - Se le llama cuando el MIDlet se destruye (-> destroyed state)
  - **`public final void notifyDestroyed()`**
    - Comunica al AMS que el MIDlet ha limpiado la memoria y ha terminado.
  - **`public final void notifyPaused()`**
    - Comunica al AMS que el MIDlet está en pausa.
  - **`public final String getAppProperty(String key)`**
    - Se le llama para obtener las propiedades del MIDlet.

MIDlet: ¡Hola Mundo!

```
import javax.microedition.midlet.MIDlet;
import javax.microedition.lcdui.*;

public class HelloWorld extends MIDlet implements CommandListener {
  private Display display;
  private TextBox mainScreen;
  private Command exit;

  public HelloWorld() {
    mainScreen = new TextBox("Saludo","Hola alumnos APMOV",512,0);
    exit = new Command("Exit", Command.EXIT, 1);
  }
```

MIDlet: ¡Hola Mundo!

```
  public void startApp() {
    mainScreen.addCommand(exit);
    mainScreen.setCommandListener(this);

    display = Display.getDisplay(this);
    display.setCurrent(mainScreen);
  }
  public void pauseApp() {
  }
  public void destroyApp (boolean unconditional) {
  }
  public void commandAction(Command c, Displayable s) {
    if (c == exit) {
      display.setCurrent(null);
      destroyApp(false);
      notifyDestroyed();
    }
  }
}
```

MIDlet: ¡Hola Mundo!

- Fichero .JAD

```
MIDlet-Name: HelloWorld
MIDlet-Vendor: UC3M_APMOV
MIDlet-Version: 1.0
MIDlet-Jar-Size: 1204
MIDlet-Jar-URL: HelloWorld.jar
MIDlet-1: HelloWorld, HelloWorld.png, HelloWorld
MicroEdition-Configuration: CLDC-1.1
MicroEdition-Profile: MIDP-2.0
```

MIDlet: ¡Hola Mundo!

- Manifiesto

```
MIDlet-Name: HelloWorld
MIDlet-Vendor: UC3M_APMOV
MIDlet-Version: 1.0
MIDlet-1: HelloWorld, HelloWorld.png, HelloWorld
MicroEdition-Configuration: CLDC-1.1
MicroEdition-Profile: MIDP-2.0
```

Java Verified™

- Si queremos que nuestras aplicaciones las utilicen terceros es importante pasar por un proceso de verificación, si es una aplicación Java ME será el Java Verified™.
  - Importante si queremos darla de alta en las tiendas de aplicaciones más populares.

- Si se concluye exitosamente el proceso de verificación, la aplicación estará firmada y se podrá publicitar con el logo Java Powered.

Imagen obtenida de http://javaverified.com/

Java Verified™: Tests

- Los tests que se le pasan a las aplicación están documentados para que el desarrollador los realice previamente:
  - *"Unified Testing Criteria for Java(TM) Technology-based Applications for Mobile Devices"*
- Estos test se dividen en categorías:
  - *Application Characteristics (AC)*
  - *Stability (ST)*
  - *Application Launch (AL).*
  - *User Interface (UI)*
  - *Localization (LO)*
  - *Functionality (FN)*
  - *Connectivity (CO)*
  - *Personal Information Management (PI).*
  - *Security (SE)*
  - *Retesting (RE)*

Java Verified™: Ejemplos tests

ST2	Power Consumption (Observation Only)	
Full Description The application does not consume battery excessively.		
Steps to conduct the test 1. Check that the terminal's battery is full 2. Perform the tests described in the present document (Java Verified Unified Testing Criteria) 3. Check the battery level after finishing the tests 4. Verify that the battery level has not decreased radically. 5. Record the result of the observation in the test report.	*Expected result* - The battery level has not decreased radically - A decrease of 20% of the initial charge is acceptable.	
Notes - This is not a requirement but an observation, i.e. the result of this test does not have an effect on the overall pass/fail verdict, but will be documented in the test report - The goal of this observation is to draw attention to and spot potential issues with battery consumption early on	*Exceptions*	
PASS □	PASSED WITH EXCEPTION □	

Tabla obtenida de http://javaverified.com/

Java Verified™: Ejemplos tests

AL2	Application start up
Test Description Application must start properly in 25s.	
Steps to conduct the test 1. Find the application icon and select it 2. "Press a button" on the device to launch the application 3. Observe the application launch In the timeline defined 4. The application should have displayed a main menu or interactive menu such as language selection screen where the use of the application can be started 5. Use some of the application features	*Expected Result* -The application starts in 25s or less, this is the time between steps 2 and 4 -No error messages are displayed -The application appears to function properly
Notes If launch time errors occur, corresponding messages must be reported by the test house in the test report. This test does not take into consideration the different screens displayed between the "button press" and the display of the main menu of the application. For example branded splash screen.	*Exceptions*
PASS ☐	FAIL ☐ PASSED WITH EXCEPTION ☐

Tabla obtenida de http://javaverified.com/

Referencias

- Java ME:
  - "Mobile Phone Programming and its Application to Wireless Networking". Fitzek, Frank H. P. and Reichert, Frank (Editors). (L/D 621.396.4) (Chapter 3).
  - "J2ME : Java 2 micro edition : manual de usuario y tutorial". Froufe Quintas, Agustín. (L/S 004.438 JAVA FRO).
  - http://download.oracle.com/javame/
- Java Verified:
  - http://www.javaverified.com