Dancing While Gaming—Multitasking VMs (MVM) on Real Devices

Jae Hyou Lee
Samsung Electronics
www.samsung.com

Yaniv Shani
Sun Microsystems, Inc.
www.sun.com

TS-3742

Goal

Deeper understanding of Multitasking VM implementation on real devices

Project Highlights

- A joint collaboration between Samsung Electronics and Sun Microsystems
- MVM deployment is based on Sun's JWC1.1.3
- Deploying most of the MSA JSRs
- Tested with hundreds of games and applications from leading content providers

Agenda

MVM overview

MVM in the configuration level

MVM in the profile level

- The Foreground and Background state
- Resource management
- Multitasking safety

MIDlets in MVM environment

Q&A

Agenda

MVM overview

MVM in the configuration level

MVM in the profile level

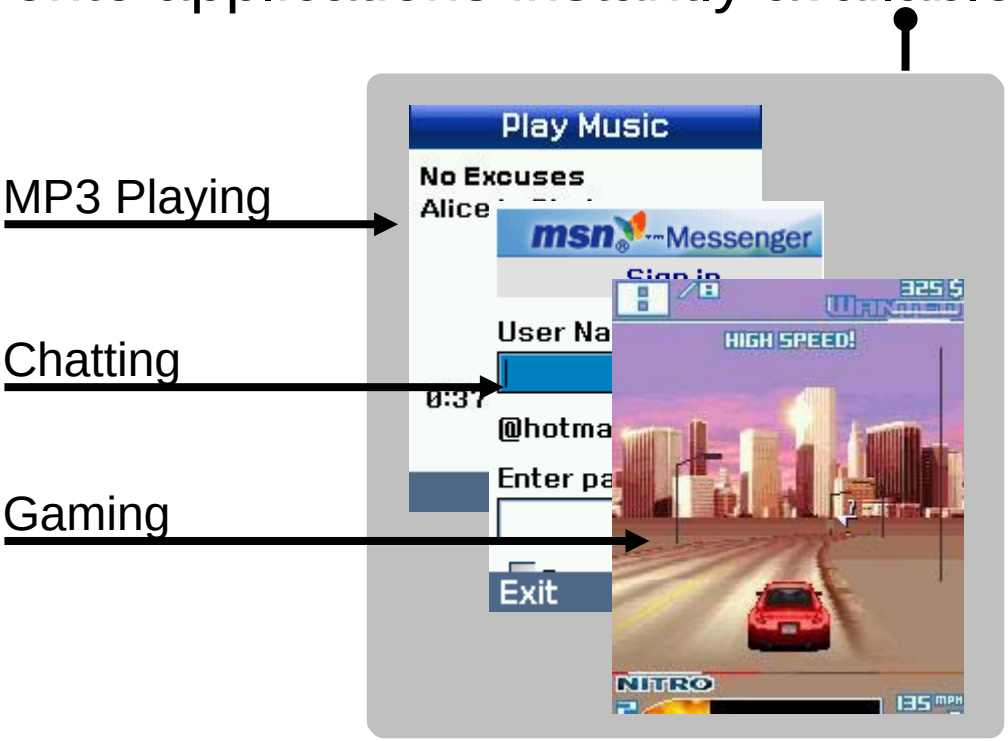
- The Foreground and Background state
- Resource management
- Multitasking safety

MIDlets in MVM environment

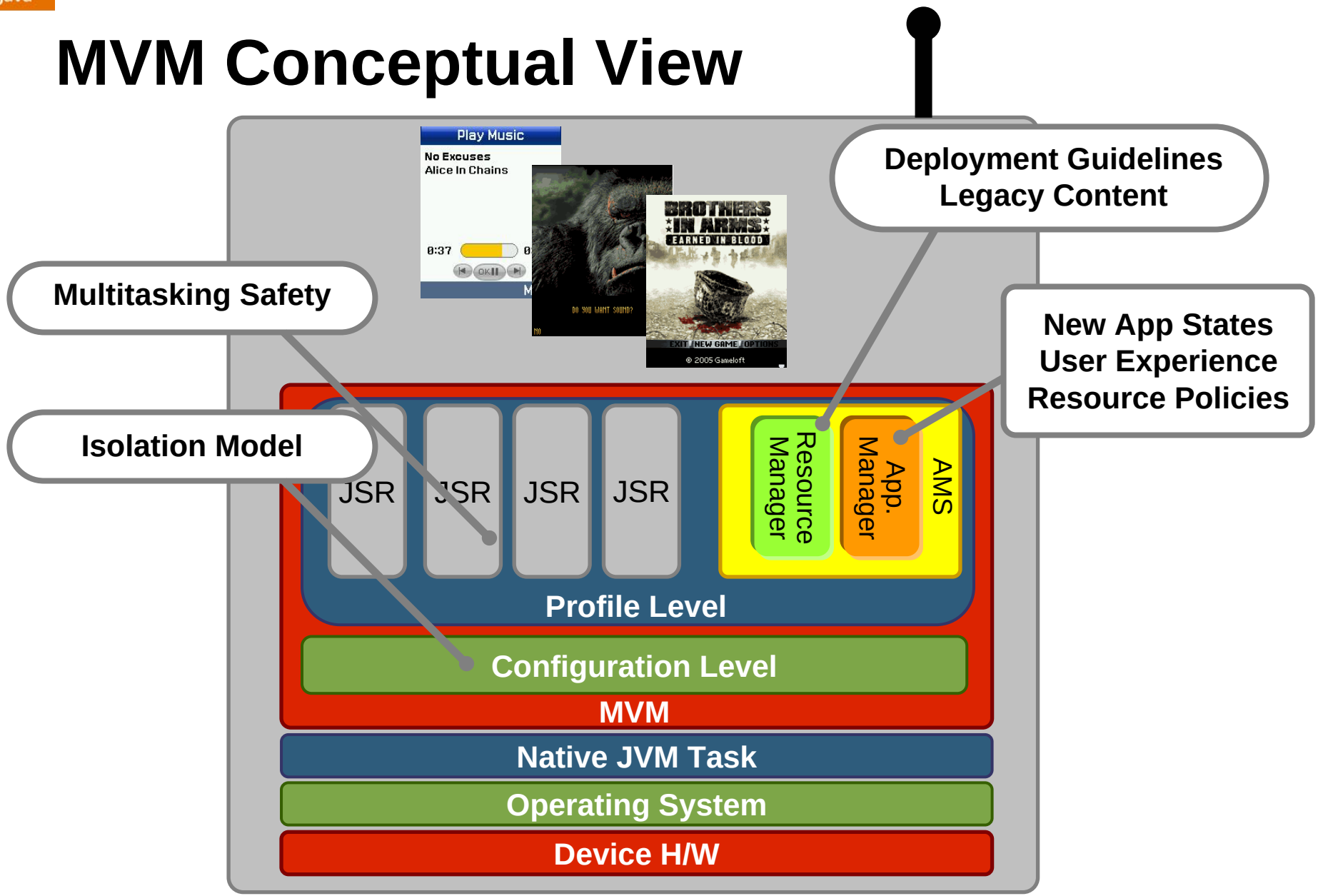
Q&A

What Is MVM?

- Running multiple applications simultaneously
- Java is always on
- Favorite applications instantly available



MVM Conceptual View



Agenda

MVM overview

MVM in the configuration level

MVM in the profile level

- The Foreground and Background state
- Resource management
- Multitasking safety

MIDlets in MVM environment

Q&A

Isolates and Tasks

Each application runs in its own logical Java Virtual Machine, called an **Isolate.**

Within the JVM, each isolate is represented as a **Task.** A task consists of one more Java thread of execution.

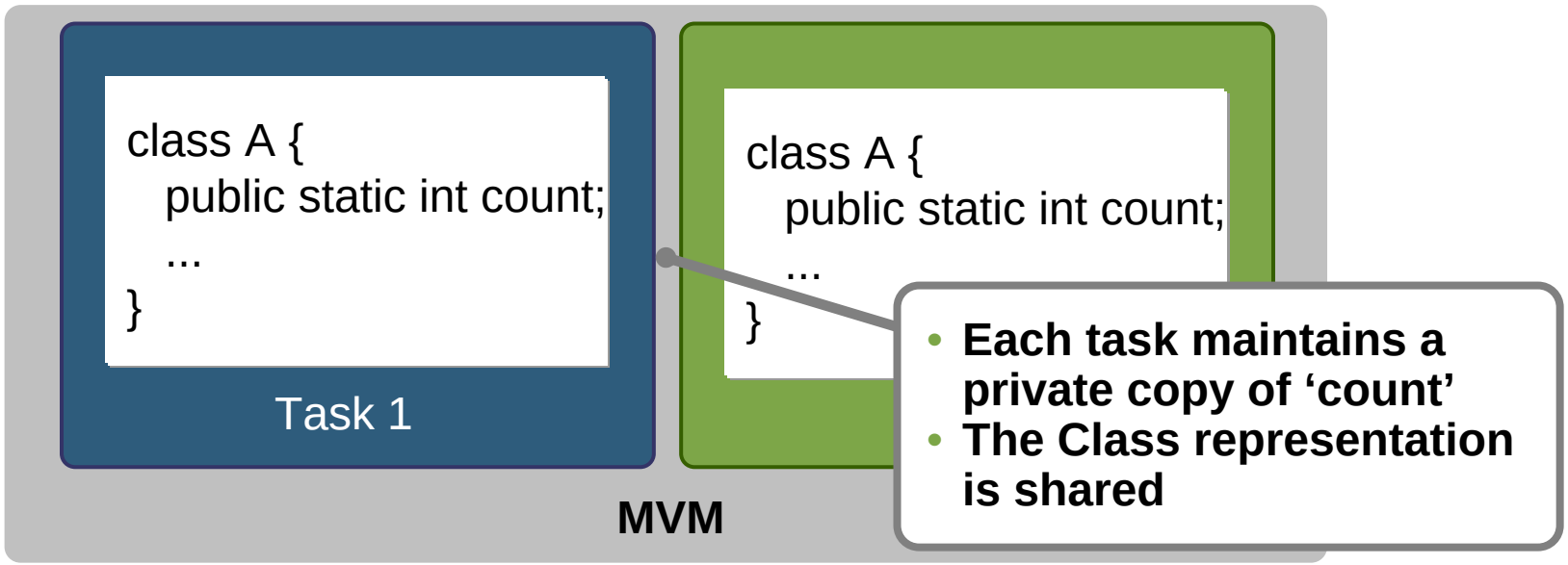
More on Tasks…

- All tasks execute in the same single OS task context
- All tasks share:
  - Constant data structure
  - Runtime support functions
- Each task encapsulates non-sharable program state

Implementation Details

Static Variable

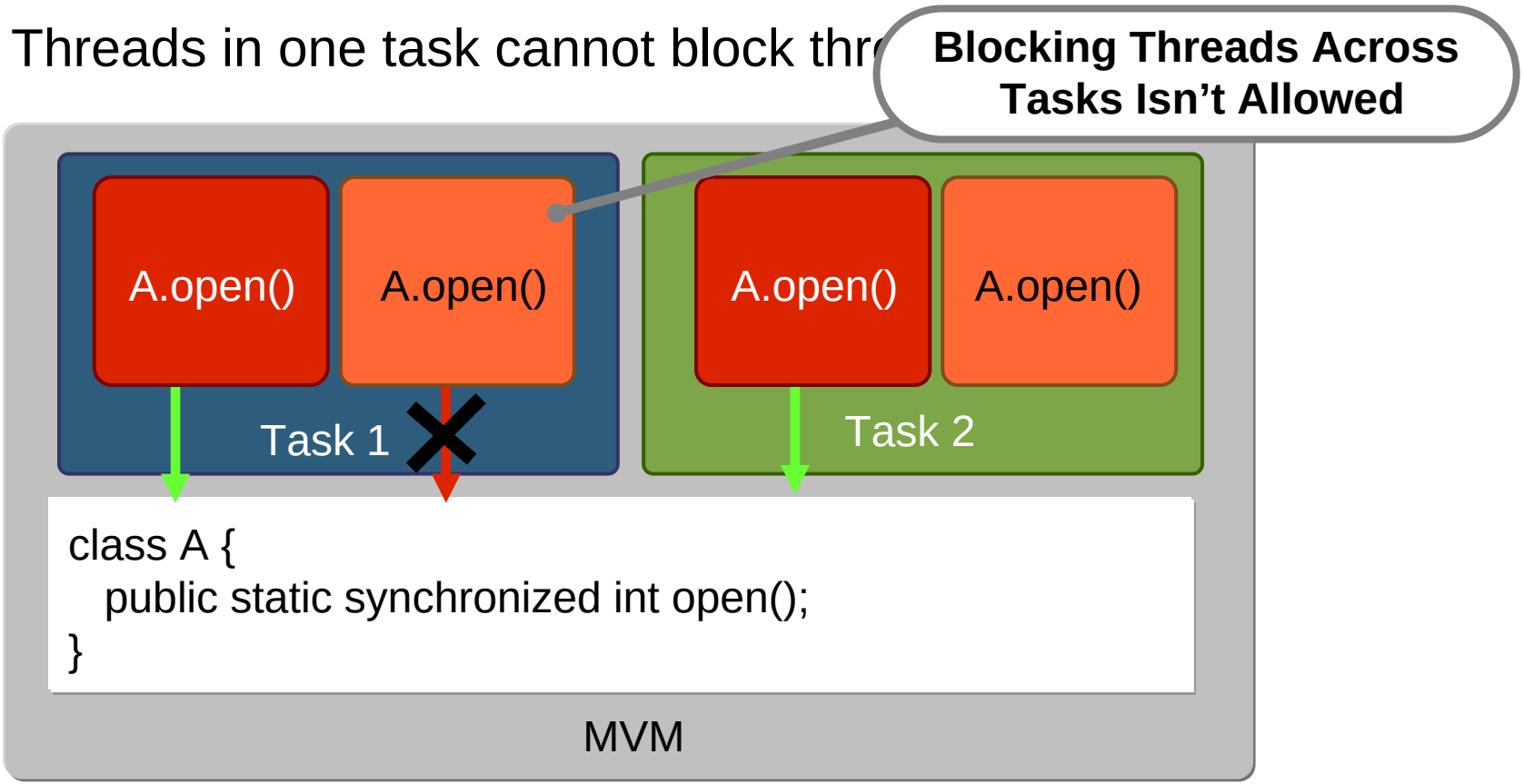
- Each task has a separate, private copy of the static variables of all the loaded Java classes
- Static variables are only global to all threads within a particular task



Implementation Details

Synchronization

- Synchronization has to be task private
- Threads in one task cannot block thr

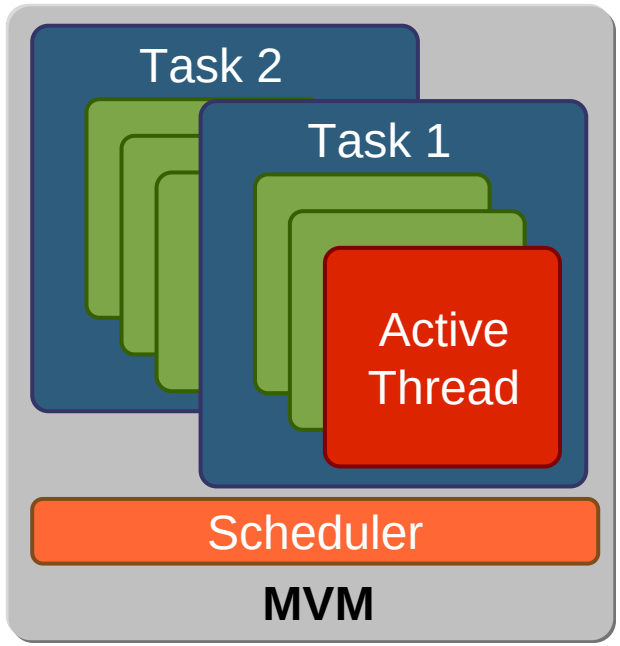


Resource Management

- The JVM is responsible for CPU time and memory resources in order to ensure that all tasks get their “fair” share
- Other resources, such as network bandwidth and communication ports are managed by the profile layer

Resource Management—Scheduling

- Tasks and thread are scheduled by the JVM
- Fair scheduling algorithm is used for task scheduling in order to prevent one task from taking disproportionate CPU time
- The ‘Isolate’ class offers a set of APIs to modify the priority levels for each task



Resource Management—Memory

- All tasks allocations are conducted from the same global heap region
- The virtual machine has a bookkeeping mechanism that accounts for each task total heap memory consumption
  - The JVM injects OutOfMemoryError as needed to inform tasks that heap space has become scarce

Agenda

MVM overview

MVM in the configuration level

MVM in the profile level

- **The Foreground and Background state**
- Resource management
- Multitasking safety

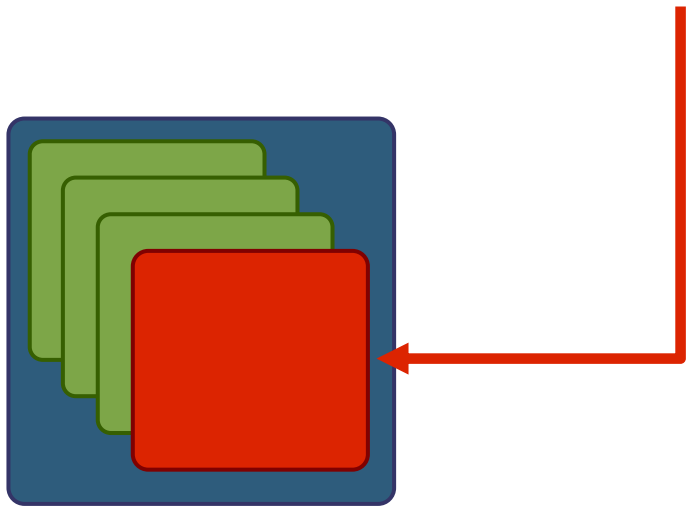
MIDlets in MVM environment

Q&A

New MIDlet State in MVM

Foreground MIDlet

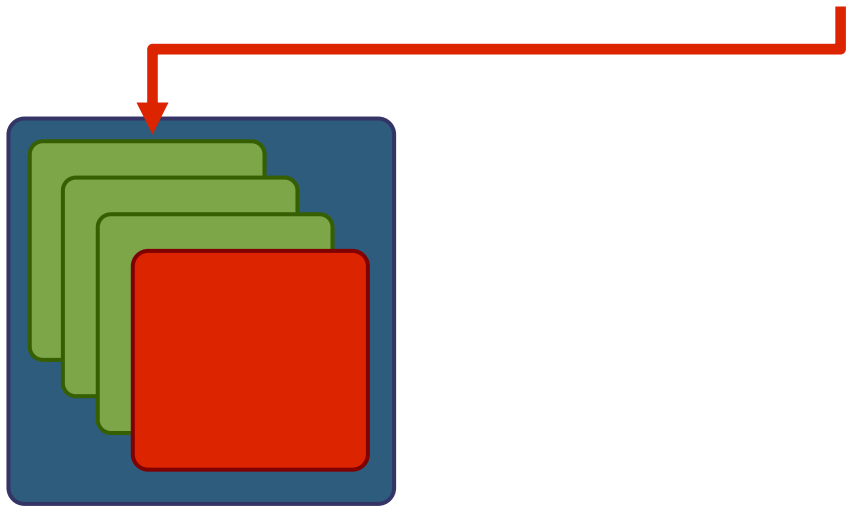
- An application is said to be in the Foreground when:
  - Its displayable controls the display
  - It handles event from the user input mechanism
- Only one MIDlet can execute in the Foreground at a time



New MIDlet State in MVM

Background MIDlet

- An application is in the Background when:
  - Its displayable does not control the display
  - It does not handle the user inputs mechanism
- Zero or more MIDlets can execute in the Background at a time



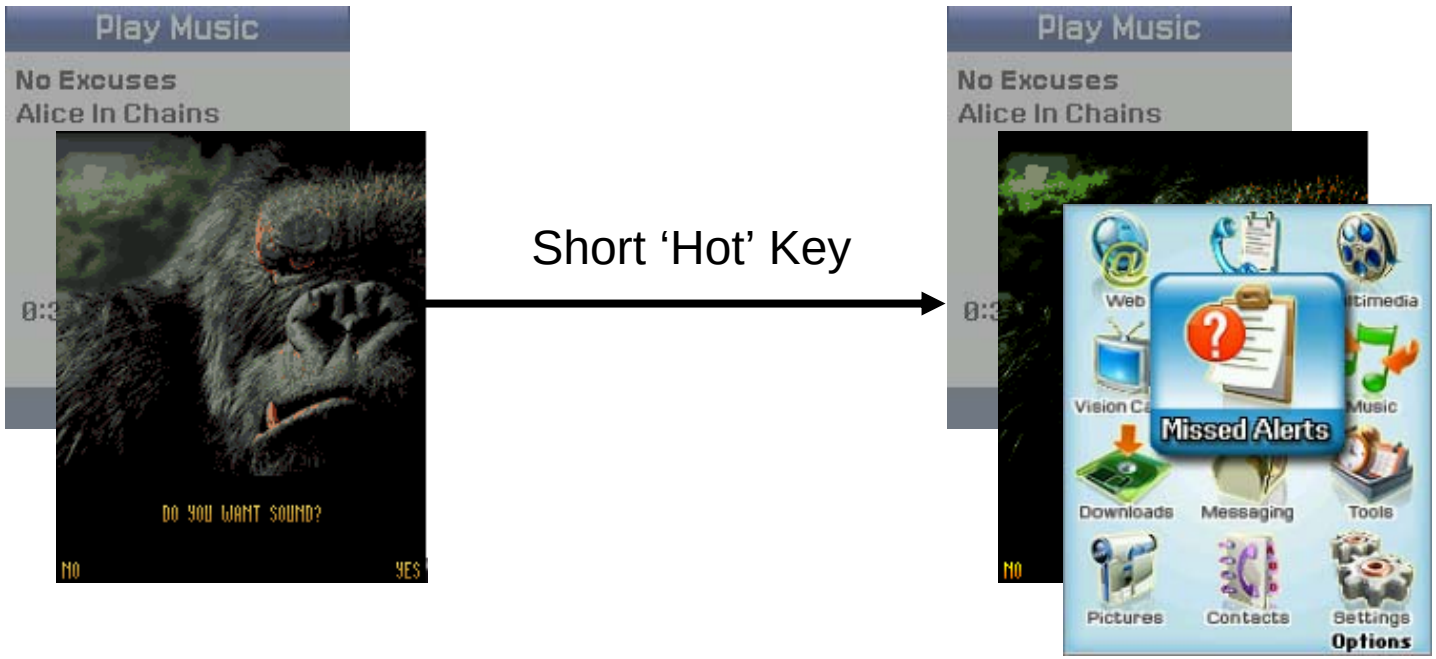
The Application Manager

- A resident MIDlet
- Executed from power on
- Lists the running applications
- A fast way to switch between running MIDlets
- Display background MIDlets alerts

Switching a MIDlet to the Background (1/3)

Short ‘Hot’ Key Press

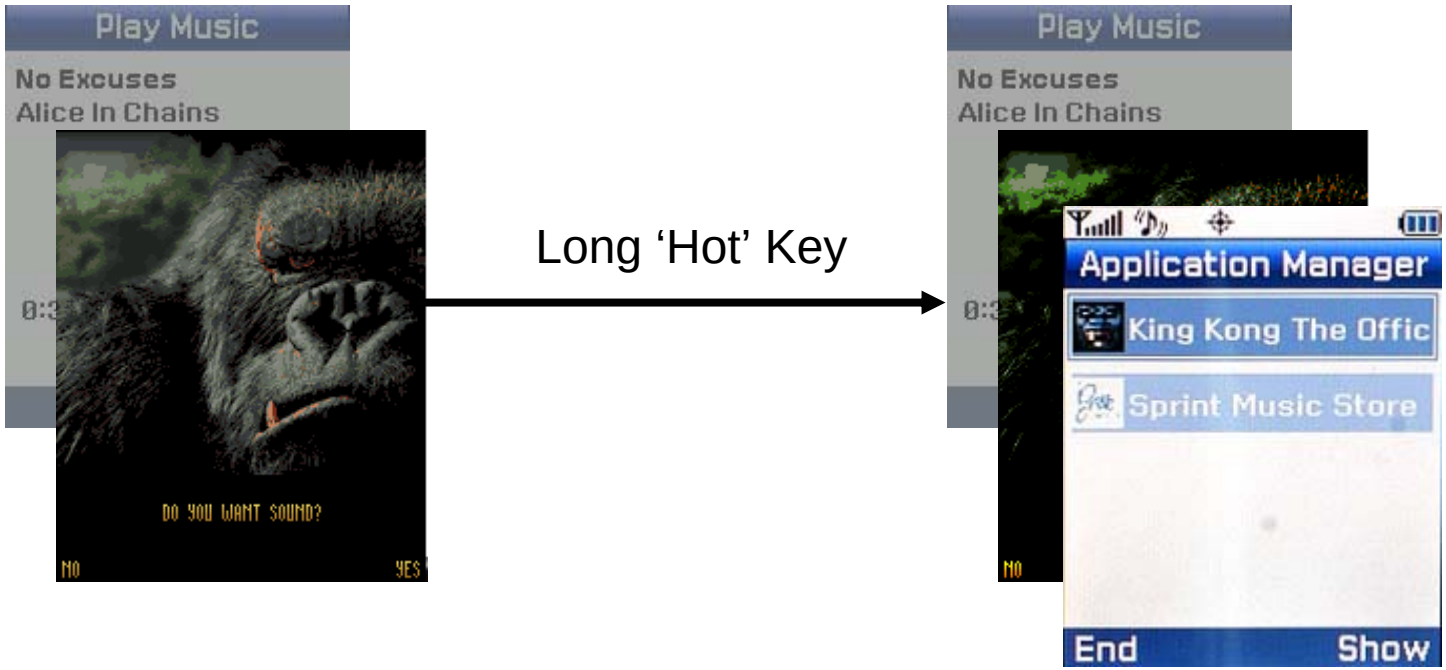
- Upon a short ‘Hot’ key press the foreground MIDlet will be placed in the background and main device menu will be displayed



Switching a MIDlet to the Background (2/3)

Long ‘Hot’ Key

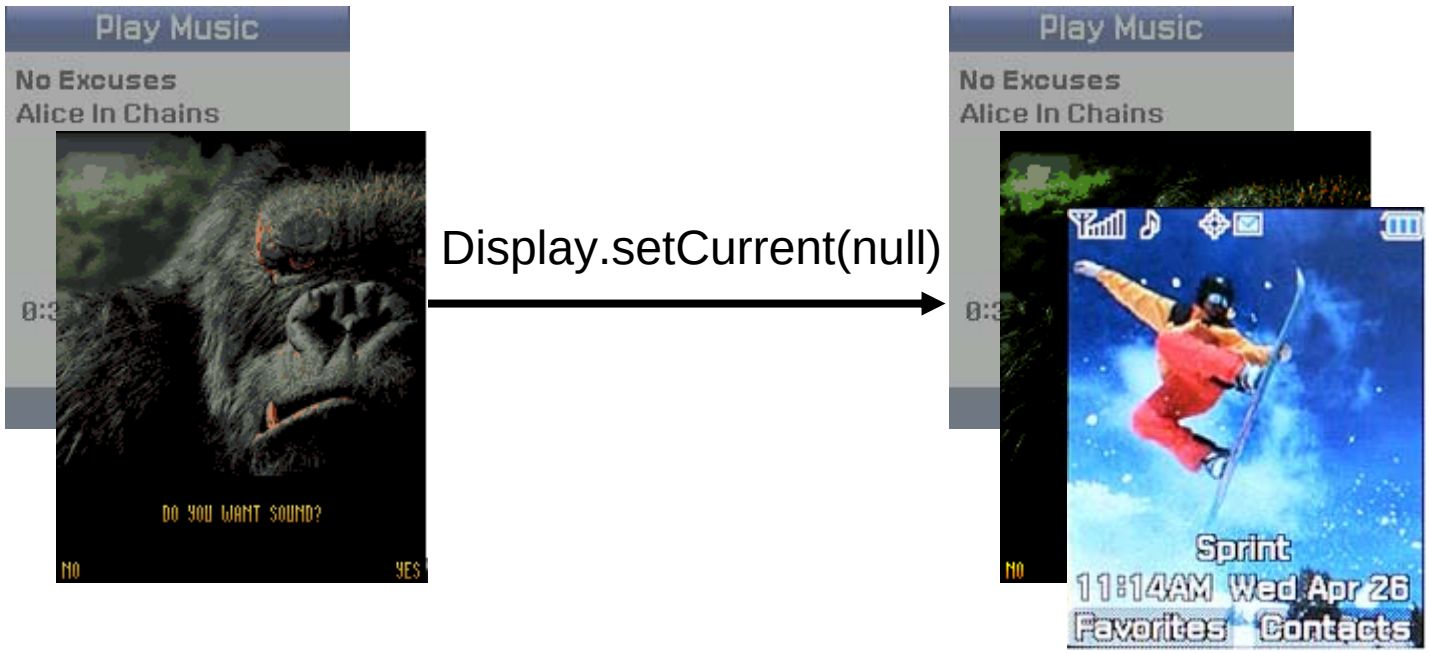
- Upon a long ‘Hot’ key press the foreground MIDlet will be placed in the background and the ‘Application manager’ dialog will be displayed



Switching a MIDlet to the Background (3/3)

Invoking Display.setCurrent(null)

- Upon calling setCurrent(null) the MIDlet will be placed in the background and the ‘idle’ screen of the device will be displayed



Interrupting the User

- Background MIDlet calls Display.setCurrent(alert)
- Background MIDlet tries to access a protected API and a security dialog is prompted
- Noteworthy event occurred in the background
  - e.g., Incoming IM while playing a game

DEMO

User Experience Demonstration

Agenda

MVM overview

MVM in the configuration level

MVM in the profile level

- The Foreground and Background state
- **Resource management**
- Multitasking safety

MIDlets in MVM environment

Q&A

Managing Resources

- In a single-tasking environment, the MIDlet has access to all of the available resources
- In a multitasking environment, MIDlets have to compete for available resources

Managing Resources

- Mechanisms
  - Reservation
  - Quota
  - Revocation
- Policies
  - Fixed partition
  - Open for competition
  - Special cases

Resource Management Mechanisms

Reservation

- A reservation mechanism sets aside a certain amount of a resource for a MIDlet and keeps it available for that MIDlet and that MIDlet alone

Resource Management Mechanisms

Quota

- A quota mechanism permits a MIDlet to allocate up to a certain amount of a resource; if the MIDlet attempts to allocate more resources than its quota, the attempt fails, even if resources are actually available on the system

Resource Management Mechanisms

Revocation

- The system can revoke a resource from a MIDlet and give it to another MIDlet
- Resource revocation examples:
  - CPU cycles
  - Display access
  - Audio playback
- Resource revocation doesn't always have a dramatic effect on the MIDlet's execution

Resource Management Policies

Fixed Partition

- Initial amount of resources are reserved before MIDlet launch
- Resource availability validation during MIDlet's startup
- Ensure consistent MIDlet behavior

Resource Management Policies

Open for Competition

- No reservation
- Resource are acquired and reclaimed at runtime
- May lead to unpredictable behavior

Special Resource Management Policies

LCD Display

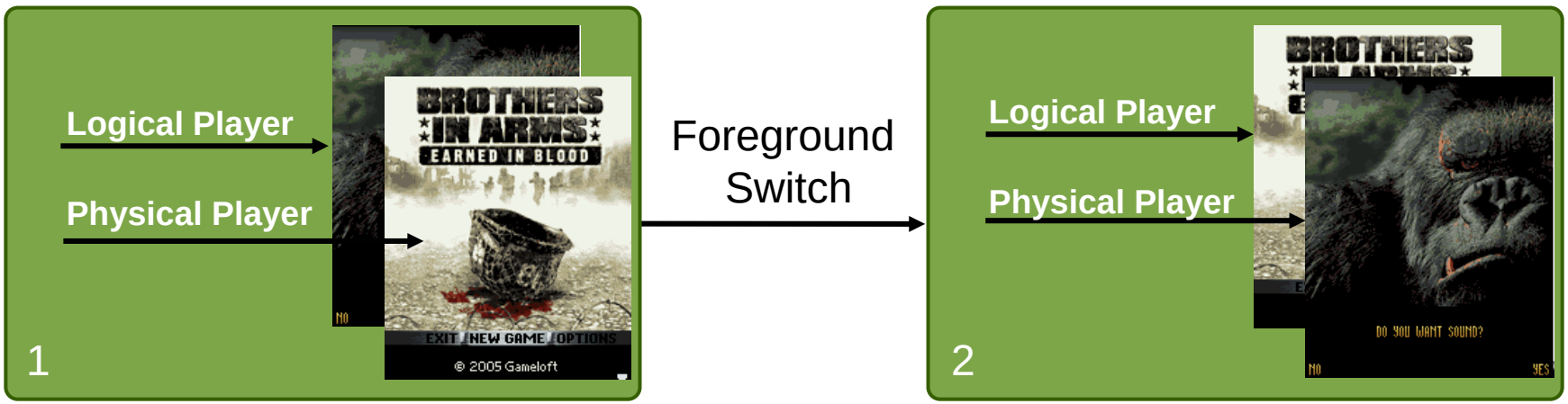
- Only the Foreground can access the LCD display and receive key inputs events
- The current displayable of a Background MIDlet is still updated

Note: vibrate() ,flashBacklight() and tone playing aren't honored when invoked from a Background MIDlet

Special Resource Management Policies

Sound (1/2)

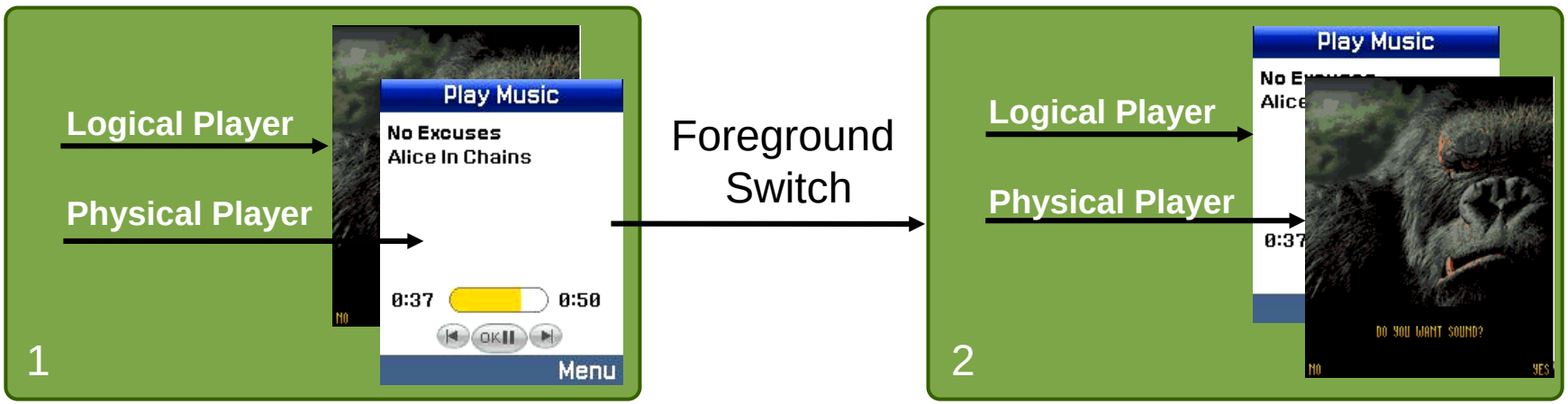
- Foreground MIDlets are assigned physical player that can access the audio device resource
- Background MIDlets are assigned a logical player that only maintains the audio state
- Upon foreground switch the player state is changed automatically



Special Resource Management Policies

Sound (2/2)

- Unique JAD file property to lock audio resources
  - Allows a background MIDlet to be audible
  - Useful for media players (e.g., the MP3 player, radio, etc.)



A Full Resource Policy Matrix

	Fixed	Open	Special
Memory	X	X	
CPU		X	X
File Descriptor	X	X	
Socket	X	X	
Display			X
Sound		X	X
Bluetooth		X	
Location		X	
PIM		X	

Guidelines for Resource Policy Selection

- Resource policy definition relies on the underlying platform capabilities and requirements
- Fixed partition policy should be used for fundamental resources that are required by the majority of the applications (e.g., memory, file, socket)
- Open policy should be used resources that are used by specific MIDlets and have limited availability (e.g., Bluetooth, location)
- Display and sound requires special handling

DEMO

Resource Policy Demonstration

Agenda

MVM overview

MVM in the configuration level

MVM in the profile level

- The Foreground and Background state
- Resource management
- **Multitasking safety**

MIDlets in MVM environment

Q&A

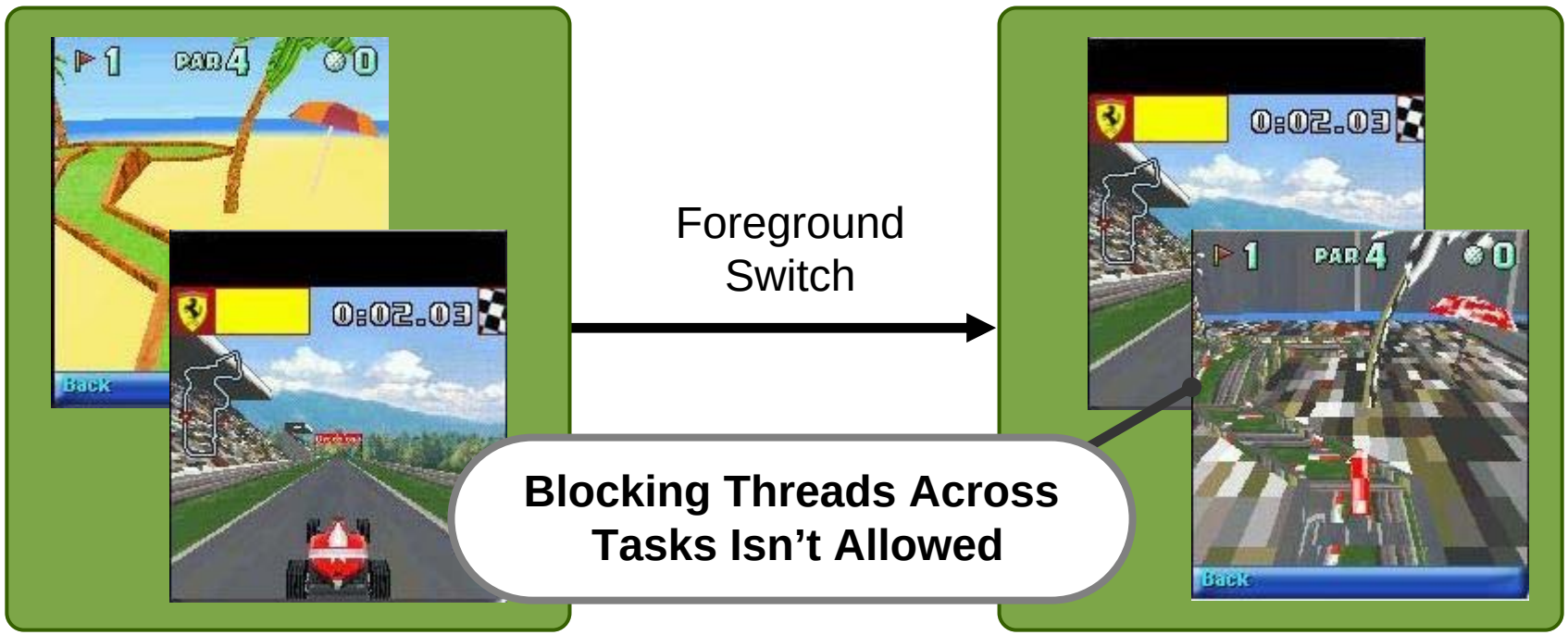
Multitasking Safety

- The native JVM task can run native code for any task, hence the code must be made aware of the task on whose behalf it is being called and possibly allocating resources
- When the task context is established, the code is considered multitasking safe
- Sun's JWC software is multitasking safe

Multitasking Safety

Implementation Guidelines

- In some cases a deployment of a profile to MVM environment requires additional development in order that the profile will execute properly



Multitasking Safety

Static and Global Variable

- At Java level, each task has a separate, private copy of the static variables of all the loaded Java classes
  - This is handled internally by the MVM
- Native code often has global data
  - Native data is shared across all tasks!
- Global native data needs to be replicated for each task
  - Held as a field of a Java object
- Proper handling for singleton depends on whether it is meant be used on a per-task basis (the default behavior for MVM), or by the entire system

Agenda

MVM overview

MVM in the configuration level

MVM in the profile level

- The Foreground and Background state
- Resource management
- Multitasking safety

MIDlets in MVM environment

Q&A

MIDlet Guidelines

MIDlet State Detection

- MIDlets developed on top of ‘Canvas’ class will receive the following notifications:
  - Canvas.hideNotify(): When switching to background
  - Canvas.showNotify(): When returning to foreground
- MIDP2.0 spec. doesn’t offer APIs for MIDlet that doesn’t use the ‘Canvas’ class facilities

MIDlet Guidelines

In the Background, Am I Paused?

- Switching to background does not pause a MIDlet by default
- Legacy content handling
  - Some MIDlets (e.g., legacy games) will not run correctly in the background
  - A new JADfile property can request that the MIDlet be paused when switched to background

MIDlet State Detection

How to Behave While Executing in Background?

- Most of the games should be suspended in background
  - Player may be inadvertently killed
  - Free up resource that aren't in use
  - No need to continue painting when nobody sees the results
- Make sure to attract the users attention when something interesting happens in the background

MIDlet Guidelines

Resource Awareness

- Return resources when you are done with them, don’t wait for destroyApp()
- Expect and detect failures in resource acquisition
- Help the users to overcome resource acquisition failure
- Expect to see new JAD file properties for resource policy indication (heap size, special sound policy, etc.)

Boot-time MIDlets

- Launch in the background at boot time
- Bring to foreground upon arrival of incoming event
- Examples:
  - MMS, SMS, IM, Mail clients
  - Stock, weather, news

Summary

- MVM in the configuration level:
  - Isolation Model
  - JVM executes in a single native task context
- MVM in the profile level
  - Foreground and Background state
  - Resource management and policy
  - User experience
  - Multitasking safety
- MIDlets in MVM environment

Agenda

MVM overview

MVM in the configuration level

MVM in the profile level

- The Foreground and Background state
- Resource management
- Multitasking safety

MIDlets in MVM environment

Q&A

Q&A

Dancing While Gaming—Multitasking VMs (MVM) on Real Devices

Jae Hyou Lee
Samsung Electronics
www.samsung.com

Yaniv Shani
Sun Microsystems, Inc.
www.sun.com

TS-3742