JavaOne℠

Sun's 2001 Worldwide Java Developer Conference

The Java TV™ 1.0 API: Technical Overview

Jon Courtney
Java TV Specification Lead
Sun Microsystems

Purpose of This Presentation

To help you become familiar with the primary features that the Java TV™ API provides for creating applications for interactive digital television

Learning Objectives

After hearing this presentation, you will understand:

- The scope of the Java TV API
- The role of the Java TV API in interactive digital television systems
- How the major architectural elements of the Java TV API work

About the Speaker

Jon Courtney…

Led the completion of the Java TV 1.0 API specification

Represented Sun and promoted Java TV API in television standards bodies in U.S. and Europe

Currently specification lead for J2ME™ technology Personal Profile

Presentation Topics

Background and context of Java TV API

Java TV API Features

- Purpose
- Design
- Examples

Q&A

What Is Java TV™ API?

The Java TV API is a set of extensions to the PersonalJava™ application environment to enable TV-centric applications and services

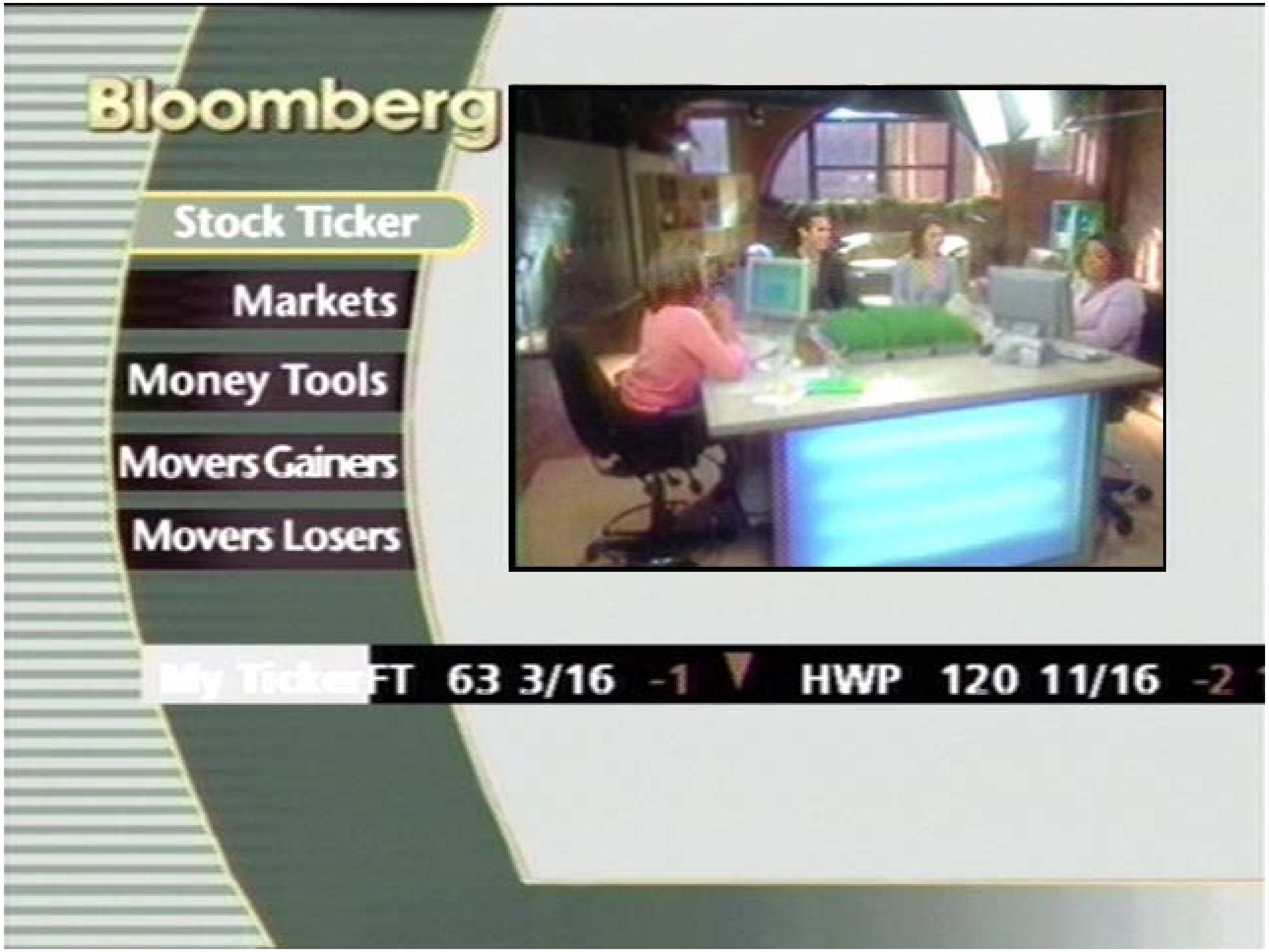
What Is the Java TV API?

Provides DTV-specific APIs

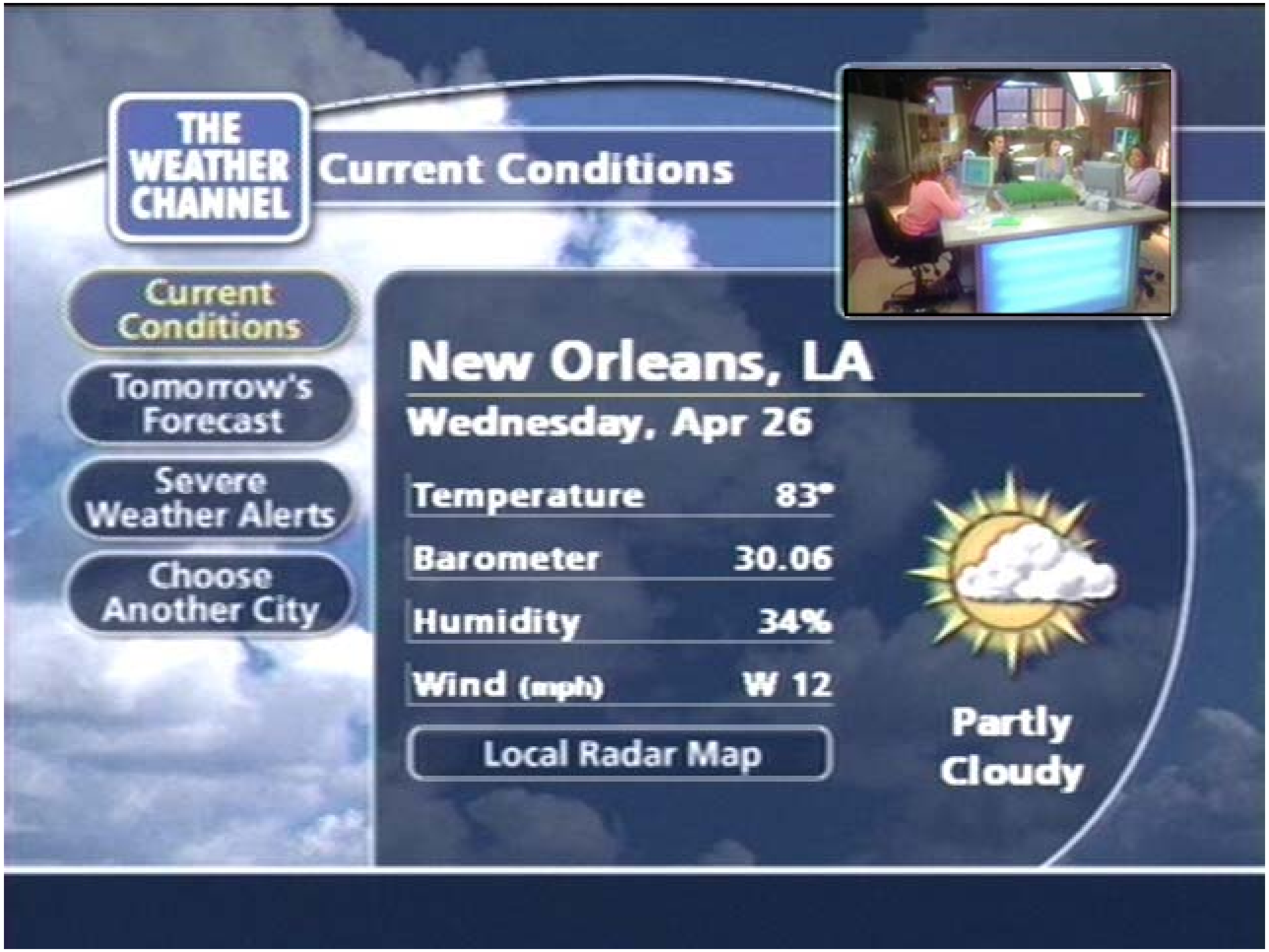
Developed through Java Community Process^{SM} initiative

Adopted in worldwide DTV standards

Example: Interactive Data Service



Example: Personalized Content



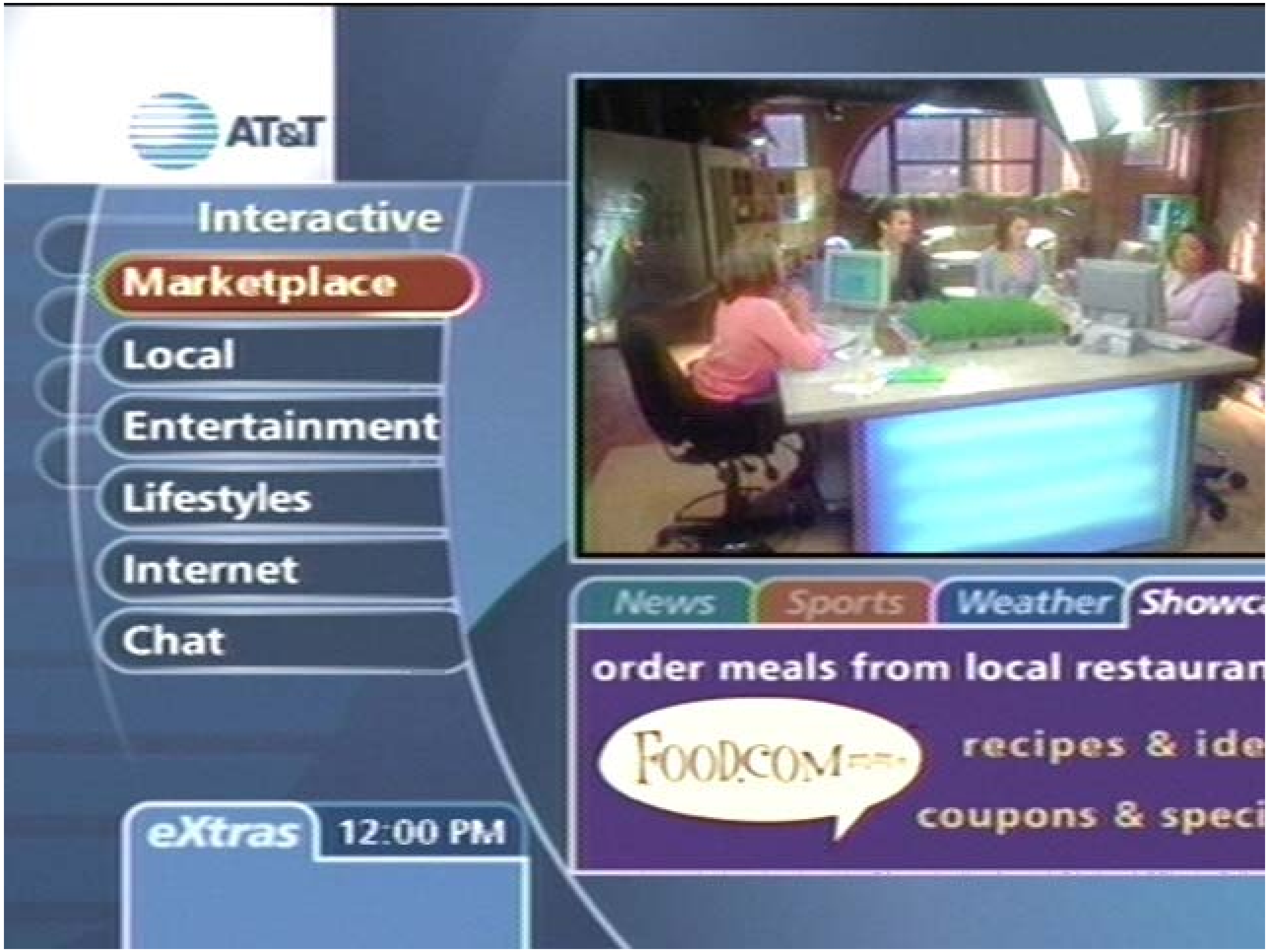
New Orleans, LA

Wednesday, Apr 26

Temperature	83°
Barometer	30.06
Humidity	34%
Wind (mph)	W 12

Partly Cloudy

Example: Premium Service Control



order meals from local restauran

recipes & ide

coupons & speci

Applications for the Java TV API

- Television Enhancement and Interaction
  - Video-synchronized, data-driven, user-interactive, adaptive presentation, animation and stream control
- Premium Video Service Control
  - PPV, Video-on-demand
- Electronic Program Guides (EPGs)
  - General purpose, service-specific, event-specific
- General Applications
  - E-mail, web browsing, e-commerce

Java TV API Expert Group

Toshiba

Matsushita

LG Electronics

Philips

Motorola/GI

PowerTV

Sony

Nokia

Open TV

@Home

OpenCable

Samsung

Digital TV Standards

- Digital Video Broadcasting (DVB)
  - 220+ organizations, 30+ countries
  - Java TV API referenced as basis of the Multimedia Home Platform (MHP) specification
- CableLabs
  - OpenCable specification to support interoperable cable set-top boxes
  - Java TV API referenced as basis of OpenCable Application Platform (OCAP)

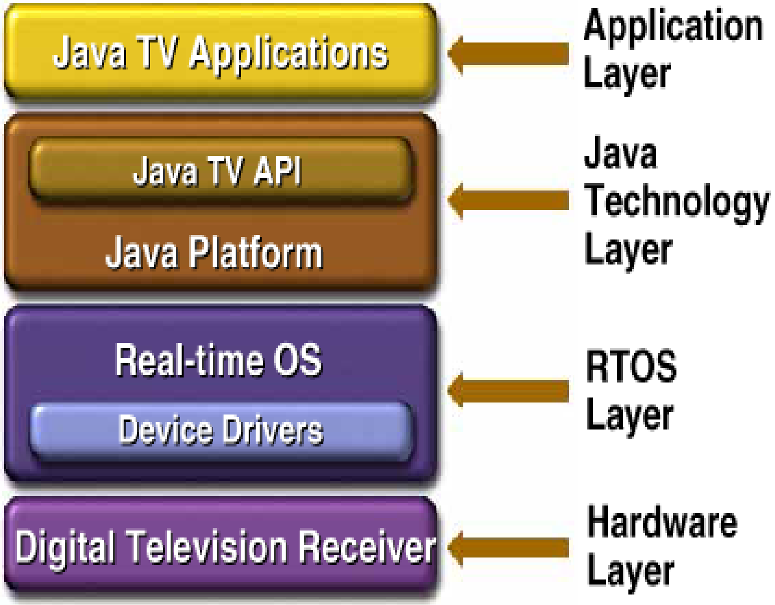
Digital TV Standards

- Advanced Television Systems Committee (ATSC)
  - Adopted by U.S. FCC, Canada, S. Korea
  - Java TV API referenced as basis of Digital Application Software Environment (DASE)
- Digital Television Industry Association (DTVIA)
  - Chinese government ministry
  - Collaborating to include Java TV API in DTV specification

Digital Television Receiver

Java™ Platform

- Virtual Machine
- Core APIs
- UI APIs
- Java TV API

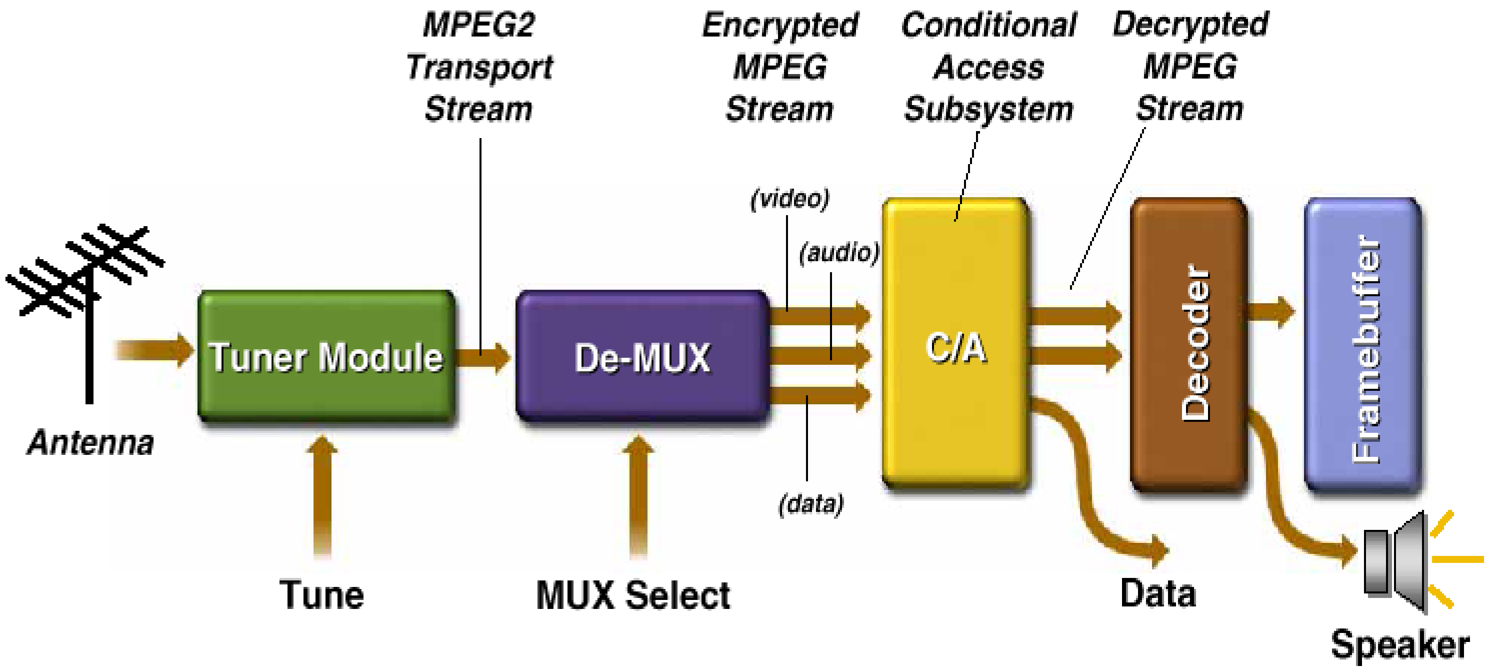


Core Java™ Platform Features

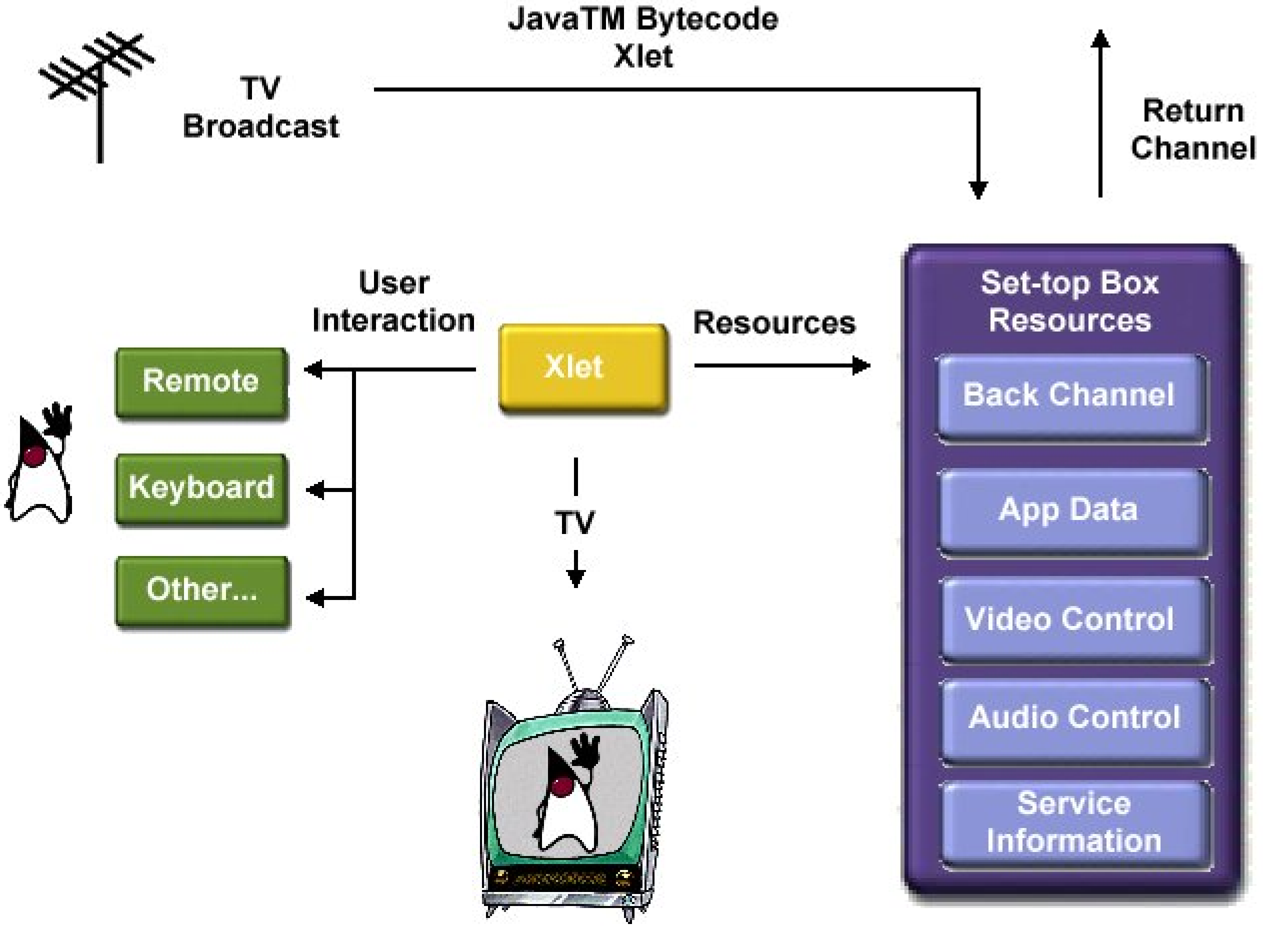
- Basic services for TV applications
  - Input/Output
    - java.io
  - Networking
    - java.net
  - Graphics and UI
    - java.awt
  - System functions
    - java.lang, java.util, java.security

Broadcast Platform

Major Hardware Components



Applications for the Java TV API

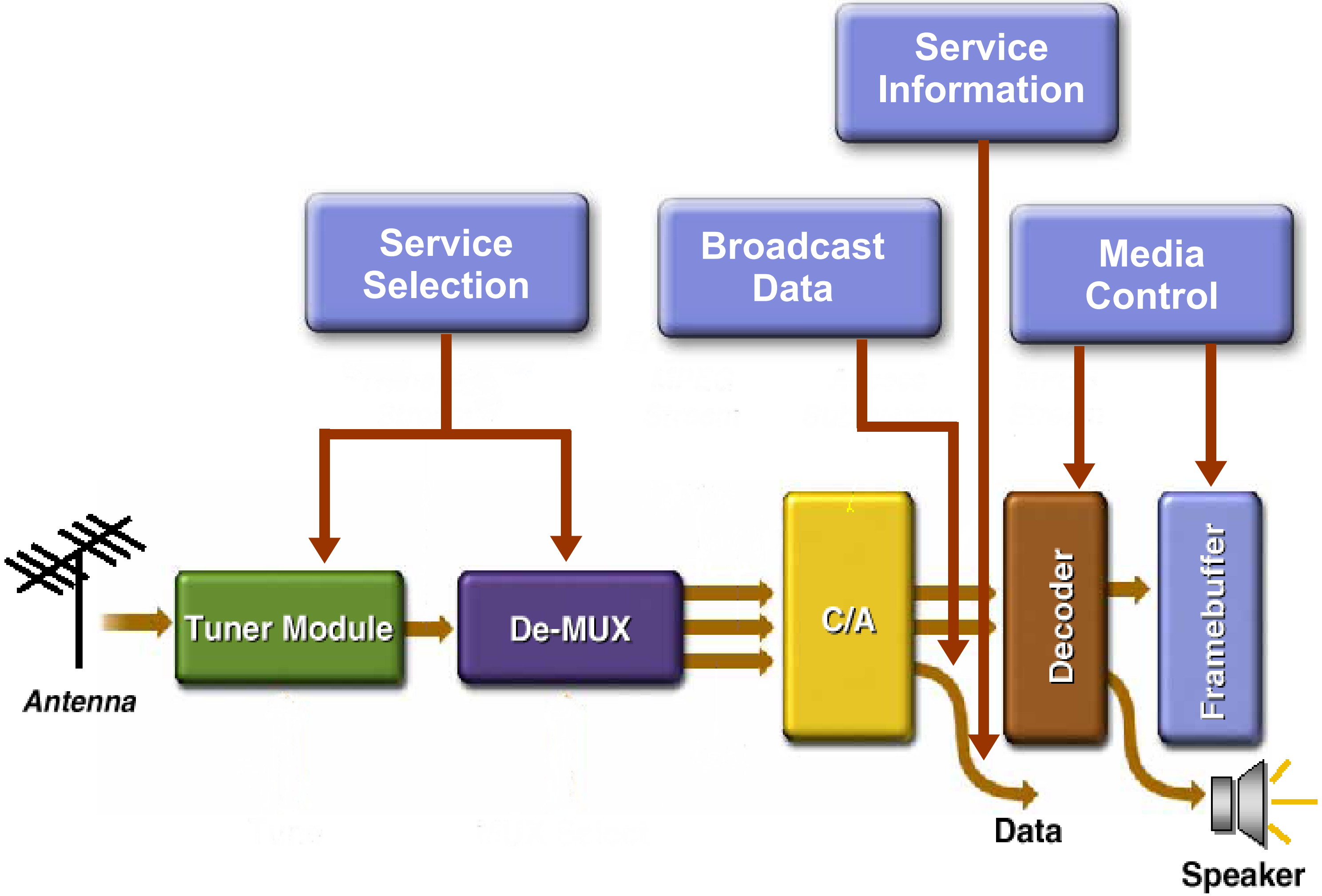


JavaOne™

JavaOne
Sun's 2001 Worldwide Java Developer Conference

Java TV API: Architecture and APIs

Java TV APIs


Service Information
Service Selection
Broadcast Data
Media Control
Antenna
Tuner Module
De-MUX
C/A
Decoder
Framebuffer
Data
Speaker

Java TV Architecture

Major API Elements

- Application life cycle
- Service Information
- Service Selection
- Broadcast Data
- Media Control

JavaOne℠

Sun's 2001 Worldwide Java Developer Conference

Java TV Architecture: Application Life Cycle Model

Application Life Cycle Model

Design objectives:

- Learn from existing application models
- Develop a model appropriate for TV

Introducing: The *Xlet*

Features:

- Ease of use for application developers
- Model separate from:
  - Window system management
  - Resource management policy
  - Application management policy
- Minimal requirements on app managers

Application Life Cycle Elements

Application Manager

Xlet interface

XletContext interface

Xlet State Machine

Application Manager

Receives and manages Xlets

Tracks state of Xlets

Can change the state of Xlets

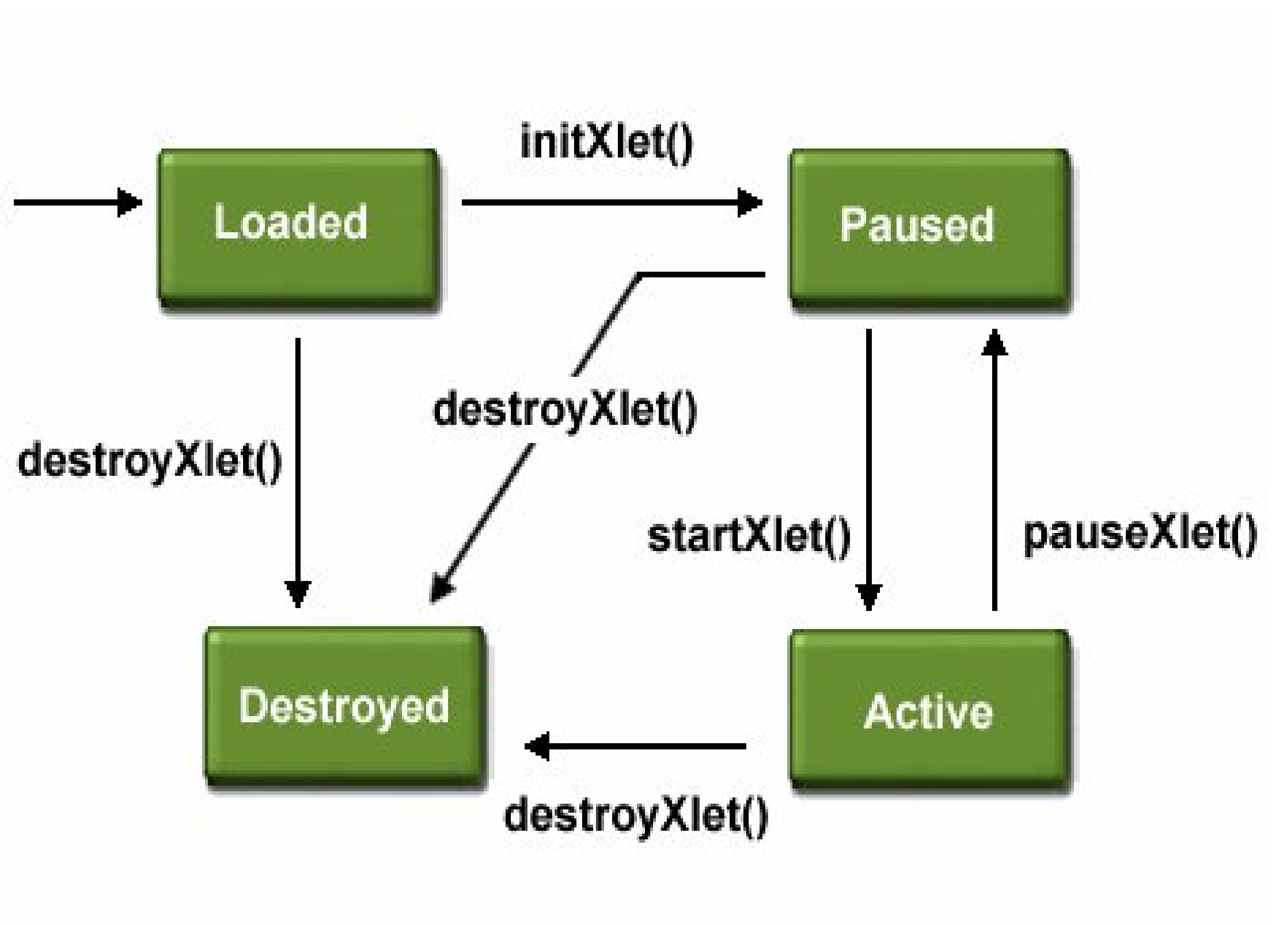
Can destroy an Xlet at any time

Application States

Four states defined:

- Loaded
  - Code is loaded, initialized
- Paused
  - Xlet quiescent, minimal resource usage
- Active
  - Xlet is executing normally
- Destroyed
  - Xlet has released resources, terminated

Application State Machine



Xlet Interface

Implemented by the Java TV application

Methods to signal state transitions

Xlets managed by Application Manager

Similar to applet model w/o UI

Xlet Interface

```
Package javax.tv.xlet;
public interface Xlet {
  void initXlet(...);
  void pauseXlet();
  void startXlet();
  void destroyXlet(...);
}
```

XletContext

Provides property interface

Used by Xlet to signal state transitions to the application manager

Xlet.initXlet(XletContext context);

XletContext

```
package javax.tv.xlet;
public interface XletContext {
  Object getXletProperty(String);
  void notifyPaused();
  void resumeRequest();
  void notifyDestroyed();
}
```

JavaOne℠

Sun's 2001 Worldwide Java Developer Conference™

Java TV API:
Service Information API

What Is a *Service*?

A collection of content for display

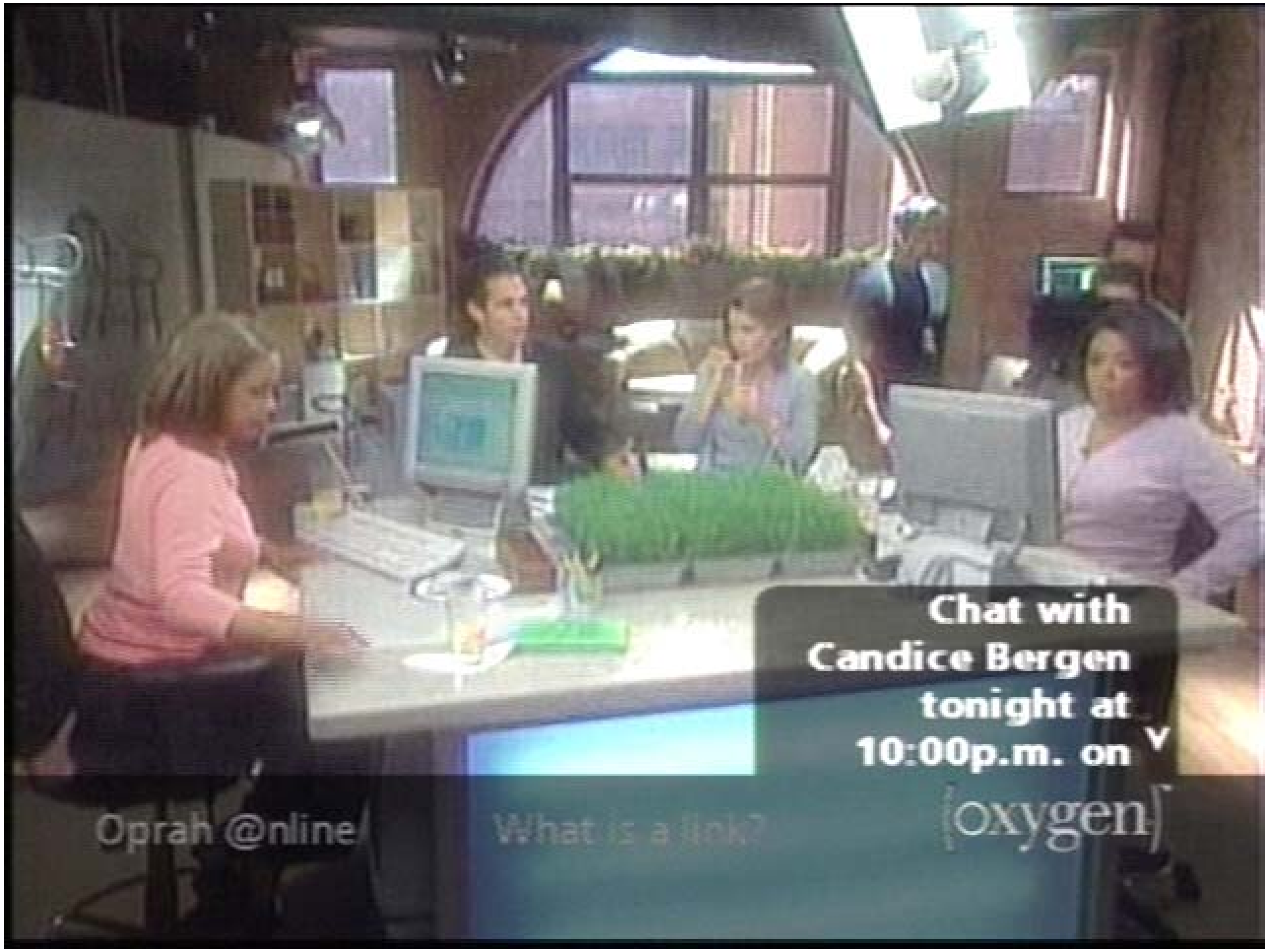
Audio/Video/Applications/Data

AKA “channel”

What Is *Service Information* (SI)?

- Data in the broadcast stream
  - Describes available services
- Populates SI Database
- Read via SI APIs

Service Information Example



Chat with Candice Bergen tonight at 10:00p.m. on

Service Information API

- Features
  - Protocol independent
  - Storage and delivery independent
  - Extensible for new SI types
  - Cached and non-cached access
    - Synchronous and asynchronous access
  - Service discovery

Service Information API

Three “views” of service information:

- Navigation package
    - Traversing through hierarchical SI data
- Guide package
    - EPG support
        - Program schedules, events, rating info
- Transport package
    - Exposes SI delivery mechanisms

Retrieving Service Information

Database cannot cache all SI data

High latency in accessing data not in cache

Inconvenient for programs to block while waiting for data

Asynchronous SI Retrieval

Asynchronous retrieval mechanism permits applications to queue requests and continue execution

Asynchronous data access methods prefixed with ‘retrieve’:

```
retrieveProgramEvent(...)
```

Asynchronous SI Retrieval

Retrievable data types extend SIRetrievable interface

Interface SIRequestor implemented by applications to receive data

```
void
notifySuccess(SIRetreivable[])

void notifyFailure(...)
```

Asynchronous SI Retrieval

SIRequest objects returned by asynchronous retrieval calls

```
Boolean cancel();
```

Example:

```
SIRequest
retrieveProgramEvent(Locator,
SIRequestor);
```

Service Information Manager

- SIManager
  - Main point of access to SI database
  - Generates events describing SI updates
  - Reports available services
  - SI filtering operations

SIManager API

```
Package javax.tv.service.navigation;

public class SIManager {

   ServiceCollection
   createServiceCollection(ServiceFilter)
   ;

   Service getService(Locator);

   Transport[] getTransports();

   SIRequest retrieveSIElement(Locator,
   SIRequestor);
   ...

}
```

Service API

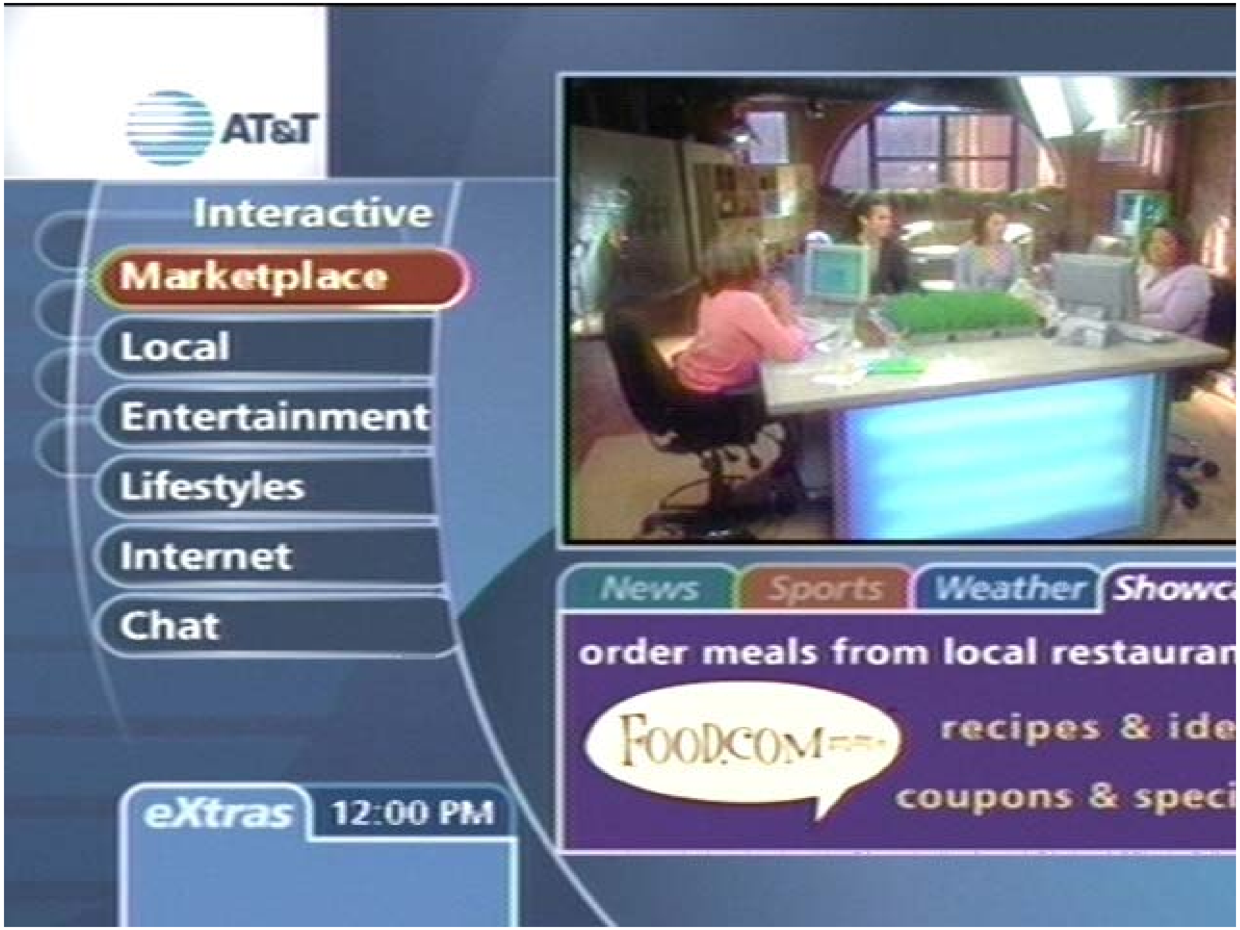
- Represents a source of content, aka “channel”
- Persistent data: name/number, locator
  - Cached, available synchronously
  - “Installed services” for bootstrap
- Asynchronous access to service “details”

ServiceDetails API

- Service meta-data
  - Represents a specific instance of a service in the broadcast
  - Reports description, program schedule, etc.
  - Reports service components and types (e.g., Audio, video, data)
- Extensible for new meta-data
- Accessed asynchronously

Java TV Architecture: Service Selection API

Service Selection Example



order meals from local restauran

recipes & ide

coupons & speci

Service Selection

Features

- Abstracts “tuning”/“channel change” operation
- Operates asynchronously
- Conditional access results exposed
- Support for multiple selection “contexts”

Service Selection: Key APIs

- ServiceContext
  - Object used to *select* a service
  - Typically maps to a physical tuner on the device
- ServiceContentHandler
  - Responsible for the presentation of a service
  - Typically related to a Java™ Media Framework (JMF) Player class

ServiceContext API

ServiceContext

- Represents an environment for presenting media and downloaded applications in a service
- Provides service selection operation

```
ServiceContext.select(Service);
```

- Reports currently selected service

ServiceContext API

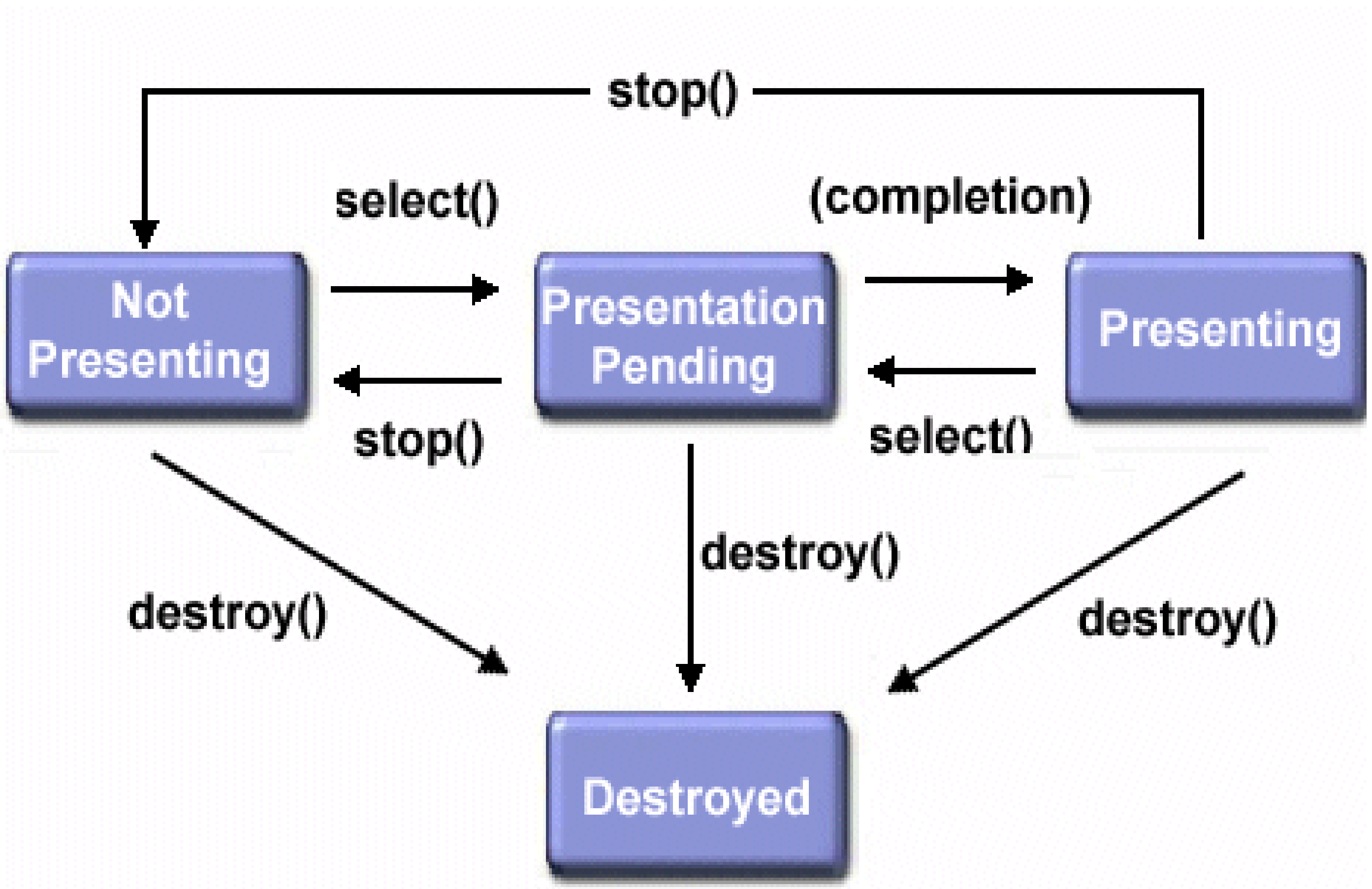
ServiceContext

- Permits management of multiple contexts
- Provides access to ServiceContentHandlers
- Signals current state via events for completion, redirection, failure

Service Context State Model

- Not Presenting
  - PresentationTerminatedEvent
- Presentation Pending
  - After select operation, before completion
- Presenting
  - Normal/AlternativeContentEvent: Content is being presented
- Destroyed
  - ServiceContextDestroyedEvent

Service Context States

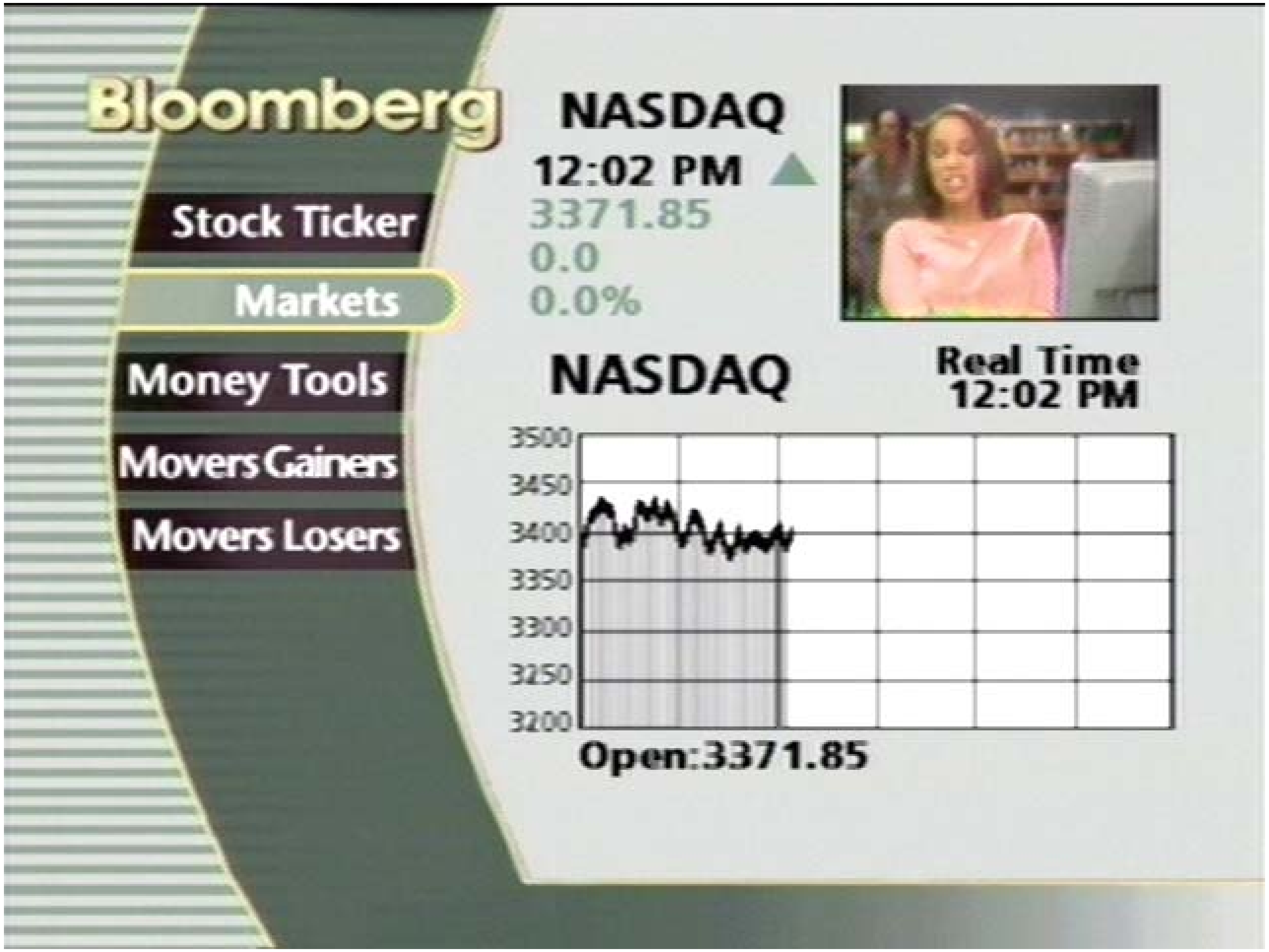


JavaOne℠

Sun's 2001 Worldwide Java Developer Conference

Java TV Architecture: Broadcast Data APIs

Broadcast Data Example



NASDAQ
12:02 PM ▲
3371.85
0.0
0.0%

NASDAQ

Real Time
12:02 PM

Open:3371.85

Broadcast Data APIs

Three modes of access:

File style access to broadcast filesystems

Push style delivery of streams

Datagram style access to broadcast IP

Broadcast File Access

- Package javax.tv.carousel
  - Provides access to bounded data in hierarchical, cyclically transmitted broadcast filesystem
  - Mappable to
    - DSMCC object carousel
    - DSMCC data carousel
    - ATVEF UHTTP

Package *javax.tv.carousel*

- CarouselFile extends java.io.File
  - Represents broadcast files
  - Familiar mechanisms from java.io package
    - FileInputStream
    - RandomAccessFile
    - FileReader

CarouselFile API

- Event notification of content changes
  - Interface CarouselFileListener
- Latency management
  - Instancing a CarouselFile notifies system to asynchronously cache file from broadcast
- Referenced via “locators” or filenames
  - Broadcast filesystem is mapped into local file name space

Streaming Data Access

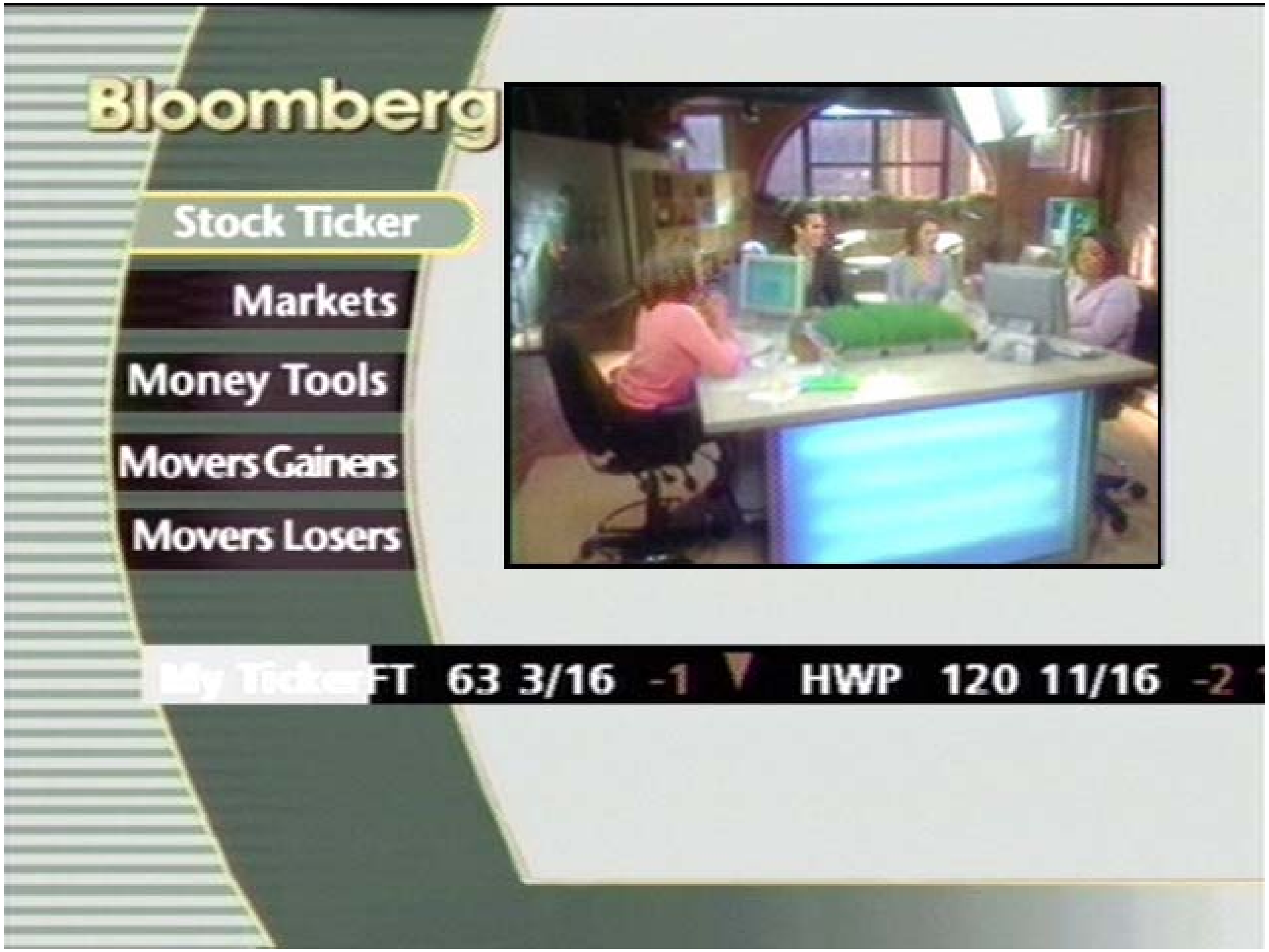
- JMF PushSourceStream Interface
  - Represents source of streaming data
  - Acquired through JMF Manager class
  - Delivers data in non-flow-controlled manner
    - Client is notified when data arrives
  - Subinterface throws exceptions for data loss

Broadcast IP Access

- Package javax.tv.net
  - javax.tv.net.InterfaceMap permits access to broadcast IP through conventional mechanisms
    - Dynamically maps handle to broadcast IP source into private local IP address
    - Access through familiar java.net mechanisms
      - DatagramSocket, MulticastSocket
    - Unicast and multicast supported

Java TV Architecture: Media Control APIs

Media Control Example

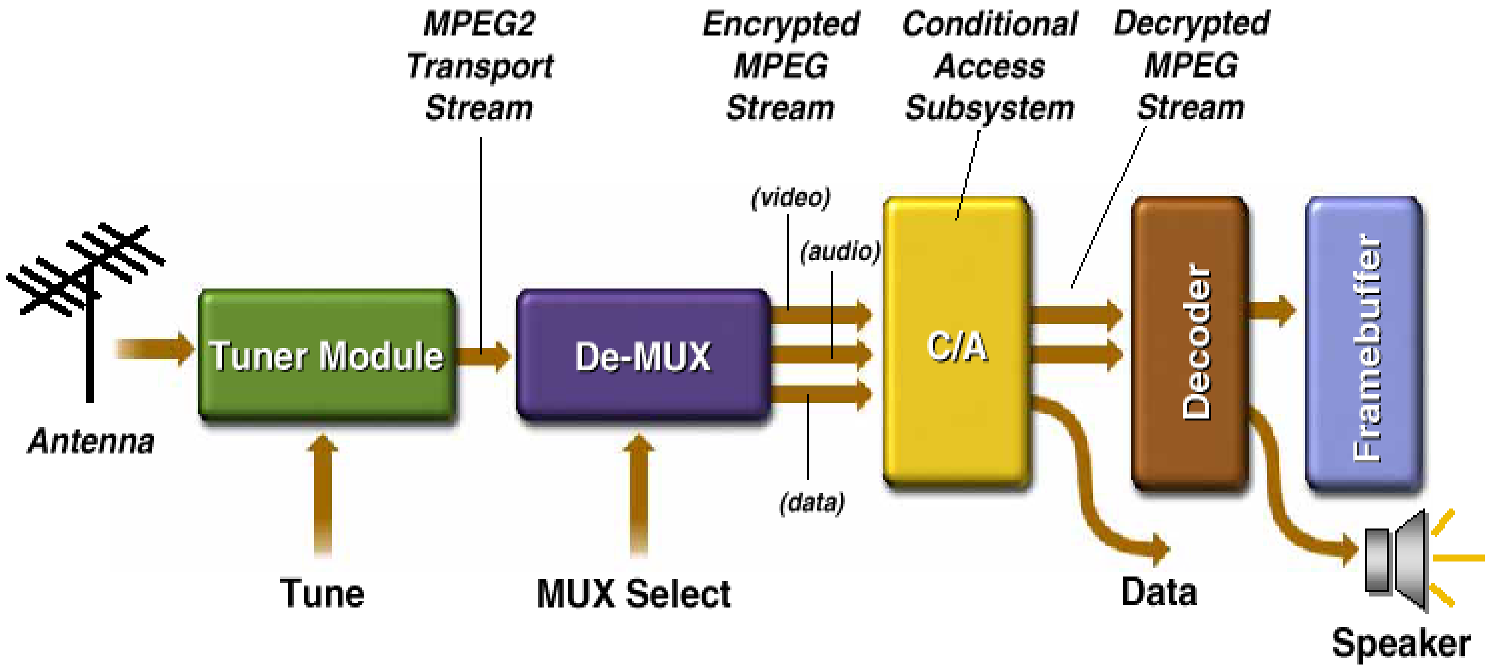


Media Control

Java Media Framework API manages pipeline

JMF Player API wraps decoder, rendering

JMF DataSource API wraps tuner and demux



JMF *Player* and *DataSource* APIs

- Representation of rendering pipeline
- Synchronization primitives
  - Media time exposed
- Players can be obtained from “presenting” ServiceContexts:
  - SC.getServiceContentHandlers();

JMF *Player* and *DataSource* APIs

- A/V control primitives
  - JMF Control objects published
  - Control set extended for TV
- Resource management mechanisms
  - Events signal state transitions
- Small framework abstracts hardware

JavaOne℠

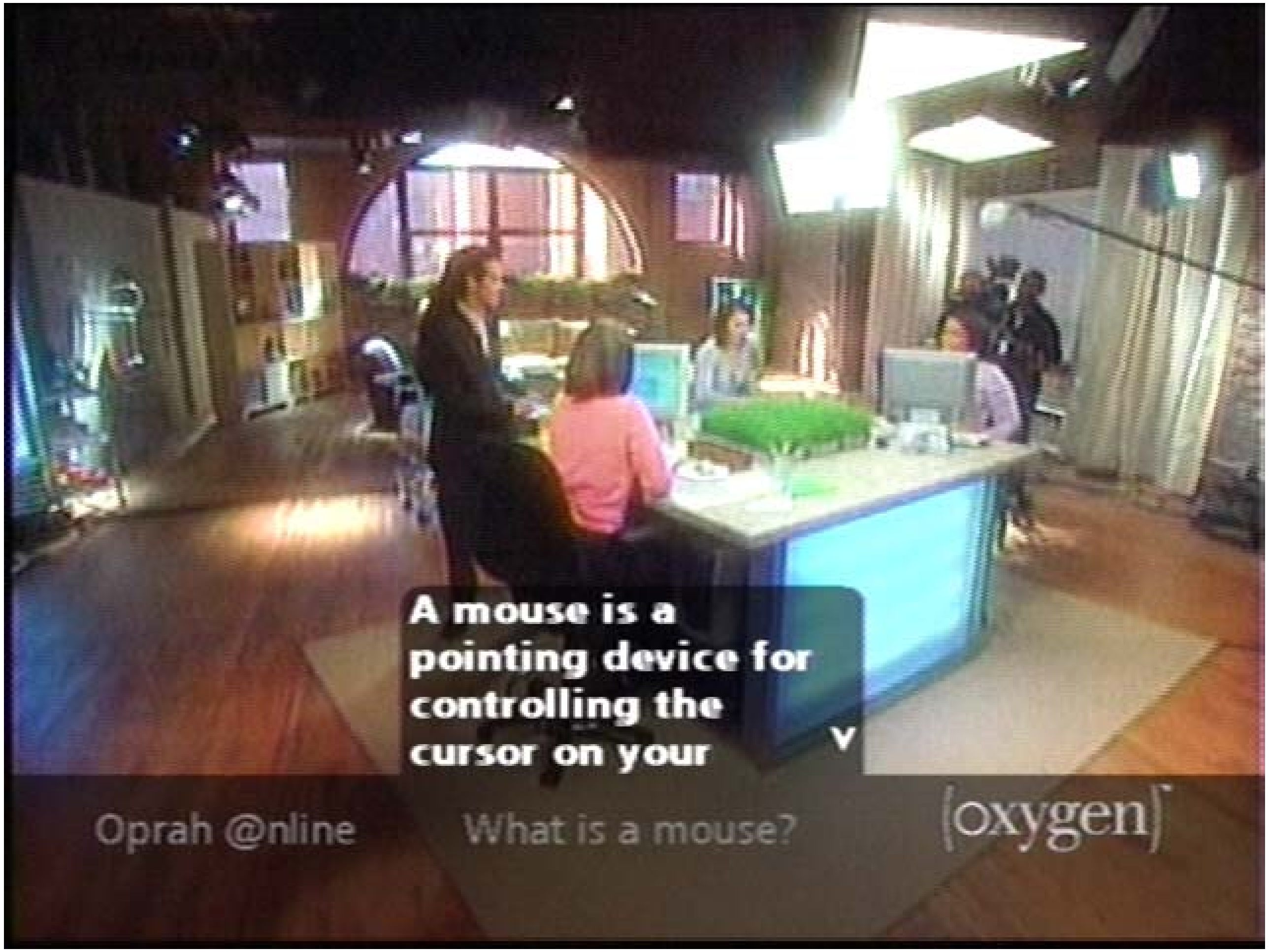
Sun's 2001 Worldwide Java Developer Conference™

Java TV API: Additional APIs

Graphics APIs

- AlphaColor
  - Subclasses java.awt.Color
  - Provides a simple alpha blending color
- TVContainer
  - Provides Xlets with a root graphics container

Alpha Blending



A mouse is a pointing device for controlling the cursor on your

JavaOne℠

Sun's 2001 Worldwide Java Developer Conference

Java TV API: Conclusion

Conclusion

The Java TV™ API…

- Provides TV-specific extensions to the PersonalJava™ application environment
- Enables development of new TV-centric applications and services
- Key component of emerging digital television systems

Resources

Java TV API website:
http://java.sun.com/products/javatv

Java TV API Reference Implementation:
http://www.sun.com/software/communitysource/javatv

DVB MHP:
http://www.mhp.org

OCAP:
http://www.opencable.com

Related Sessions and BOFs

The Java TV API: A Worldwide Standard for Interactive Television

BUS-1948, Wed., June 6, 12:15 PM – 1:15 PM

The Java TV API: Technology and Marketplace Q&A

BOF-1950, Wed., June 6, 11:00 PM – 11:50 PM

Java Technology in the OpenCable Television Environment

TS-1123, Thurs., June 7, 11:00 AM – 12:00 PM

The DVB Multimedia Home Platform, One Year On

TS-835, Thurs., June 7, 4:00 PM – 5:00 PM

JavaOne℠

Sun's 2001 Worldwide Java Developer Conference™

Q&A

JavaOne℠

Sun's 2001 Worldwide Java Developer Conference