Volantis Mobility Server

Getting Started Guide

[Document Revision 1.0]
4th October 2007

Volantis Systems Ltd.
1 Chancellor Court
Occam Road
Surrey Research Park
Guildford GU2 7YT
Surrey

Contents

Figures

Introduction

This guide will help you get started using the Volantis Mobility Server (VMS). It describes some sites that you can run 'out of the box' and access from traditional PC, desktop and laptop systems and mobile devices. It also shows how these sites are constructed and how they work.

The materials used in this guide are shipped and installed with Volantis Mobility Server.

For brevity, we use the term 'PC' in this guide to mean any conventional system including Linux and Windows workstations, personal computers and laptops.

Installation

Prerequisites

Apache Tomcat

Download and install a copy of the Apache Tomcat web application server, if you don't already have it. You can find it at `http://tomcat.apache.org/`.

The latest, stable release should be fine. The instructions in this guide are actually based on Tomcat Version 5.

Eclipse

Download and install the Eclipse IDE, if you don't already have it. You can find it at `http://www.eclipse.org/downloads/`. The latest stable release of Eclipse Classic is the best choice. The instructions in this guide are actually based on Eclipse Version 3.1.

Installing VMS

Download the installer for VMS from the web site.

After downloading VMS, start the installer. It guides you through the installation process using a series of pages.

In the **License Agreement** step, you need to accept the terms and conditions before proceeding.

In the **Choose Features** step, select full installation.

In the **Choose Repository** step, select **XML** from the drop down list. Your device and policy repository information will be stored in XML files.

In the **Choose Web Application Server** step, select **Apache Tomcat 4.1 or Later** from the drop down list. Leave the **Host** as `localhost` but change the port number if you run Tomcat on a different port.

In the **Configure ICS Server** step, **Configure ICS Host Details** page, leave the **Host** as `localhost` but change the port number if you run Tomcat on a different port.

In the **Configure ICS Server** step, **Configure ICS Caching Details** page, accept the default values.

In the **Choose Install Folder** step, take the default value provided.

Review the installation details, and complete the installation.

Product Documentation

Once installation is complete, you will be able to access product manuals. They are in the `docs` directory of the installation. Note that the product documentation refers to the commercial version of VMS, known as Multichannel Server. Most features of MCS are available in VMS.

Much of the documentation ships as on-line help and is accessed from within Eclipse.

Installing the Eclipse Plugins

VMS includes a number of Eclipse plugins for tools and documentation. These need to be copied from the VMS installation to the Eclipse installation as follows.

Copy all of the files and directories from the `plugins` subdirectory of the `eclipse` directory of your VMS installation to the `plugins` directory of your Eclipse installation. Similarly, copy all of the files and directories from the `features` subdirectory of the `eclipse` directory of your VMS installation to the `features` directory of your Eclipse installation.

Setting up the Web Applications

Below the `webapps` directory of your VMS installation, you'll find a directory called `mcs`. Copy this directory and its contents to the `webapps` directory in your Tomcat installation.

A Quick Check

At this point, you can check that everything has installed correctly by running VMS. Start your Tomcat server. Point your PC browser at

`http://localhost:8080/mcs/projects/jive/jivehome.xdime`

You should see the page shown in the top part of Figure 3. We'll look in detail at this site in *The Jive Site*, on page 9. Before that, we have a little more set up to complete.

Setting up the Eclipse Projects

It's best to stop Tomcat before setting up the projects.

Start Eclipse. Creating a new workspace will allow you to play with the VMS demonstration without affecting any other work you may be doing in Eclipse.

The Device Repository Project

You need to create a project for the device repository. From the **File** drop down, select **New** and then **MCS Project**. Give the project a name of `deviceRepository`. Uncheck the **Use Default** check box. Click **Browse** and navigate to the directory `repository/devicerepository` under your VMS installation. Click **Next**. Once again, click **Browse** and navigate to the directory `repository/devicerepository` under your VMS installation. Select the file `devices.mdpr` and click **Open**. Click **Finish** to create the new project.

Expand the project and note that devices.mdpr appears in its contents. This file contains characteristic properties for over 4,500 types of device. You can double click on it to open the tool that allows you to view its contents. The MCS on-line help, which you can access from the Eclipse Help menu, gives more details about using this tool.

The Jive Site Project

From the **File** drop down, select **New** and then **MCS Project**. Give the project a name of `jive`. Uncheck the **Use Default** check box. Click **Browse** but this time, navigate to the directory `webapps/mcs/projects/jive` under your Tomcat installation. Click **Next**. Now click **Browse** and navigate to the directory `repository/devicerepository` under your VMS installation. Select the file `devices.mdpr` and click **Open**. Click **Finish** to create the new project.

The Welcome Site Project

Repeat the process of creating a new project, but this time call it welcome and use the materials from the directory `webapps/mcs/projects/welcome`, under your Tomcat installation. Use the same `devices.mdpr` file as before.

What You Just Did

You have now set up two projects that share a single device repository, but have their own policy repositories. That means you can change the look and feel of either project without affecting the other.

You've also set up a project that allows you to look into the shared device repository to see what it contains.

We'll look at the projects in detail in *The Welcome Page Site*, on page 34 and *The Jive Site*, on page 9.

Simulating Mobile Device Access

A good way to get an initial idea about how a site will look when accessed from a mobile phone is to use the Firefox browser, together with a couple of add-ons, as a simple device simulator. This is not as accurate as using a good device emulator or a real device, but is quick and easy and is sufficient for the purposes of this guide.

You can compare the mobile and PC versions of a site by running a second, different browser concurrently. It does need to be a different browser, not just a second instance of Firefox. So on a Windows system you could use Internet Explorer, on a Mac you could use Safari and on Linux you might use Konqueror. There are lots of others of course. Pick your favourite browser that isn't Firefox for this purpose.

Getting Firefox and the Add-ons

If you don't already have Firefox installed, download it, from http://www.mozilla.com/firefox/, and install it.

To install the add ons, start Firefox and choose Add-ons from the Tools menu. Figure 1 shows the Add-ons tool as it appears in a copy of Firefox with a number of add-ons installed.

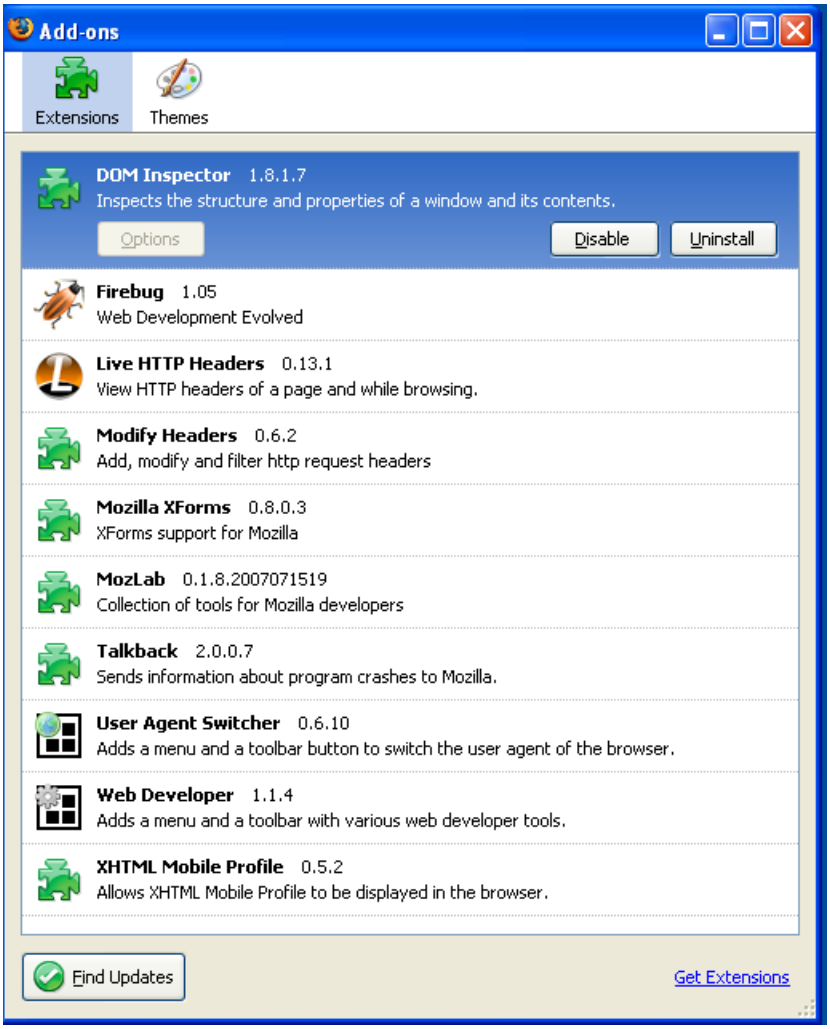

Figure 1: The Firefox Add-ons tool

You can get new add-ons by clicking on the **Get Extensions** link. This takes you to a page where you can search for add-ons.

The extensions you need are:

XHTML Mobile Profile

This extension allows Firefox to display pages marked up using XHTML Mobile Profile. Lots of mobile devices can process pages marked up this way.

User Agent Switcher

This extension allows Firefox to appear to VMS as a mobile device rather than a desk top system. The User Agent string is one of the key pieces of information that VMS uses to identify the device being used to access it. This add-on allows that string to be modified, allowing Firefox to masquerade as any device you like.

Install the add-ons.

Setting up Some User Agents

After installing the add-ons, you need to configure one or more mobile user agents. Select **User Agent Switcher**, from the Tools menu in Firefox, and then select **Options** and **Options** from the drop-downs that appear. Select **User Agents** from the dialog.

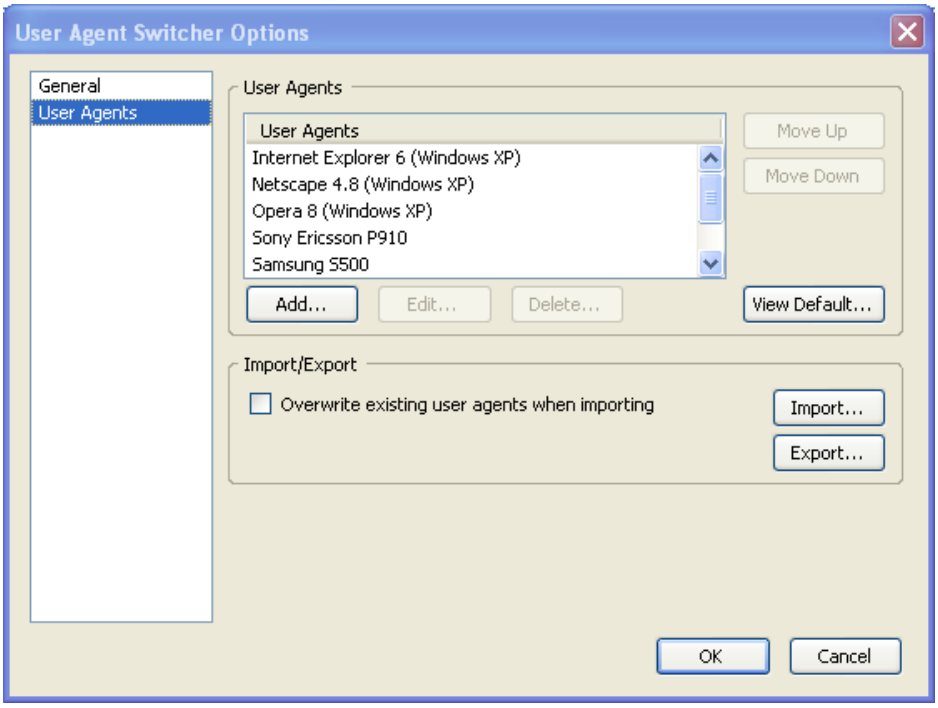

Figure 2: The User Agent Switcher Options dialog

Figure 2 shows the resulting options dialog when a number of user agents have been defined. Select Add, and fill in the resulting dialog as follows:

Description:
Sony Ericsson P910
User Agent:
SonyEricssonP910i/R3A SEMC-Browser/Symbian/3.0 Profile/MIDP-2.0 Configuration/CLDC-1.0

Note that the User Agent needs to be entered *exactly* as written above.

Click **OK**.

Add another entry, this time for the Nokia 6300, using the following values:

Description:
Nokia 6300
User Agent:
Nokia6300

Close the options dialog.

Now, when you select **User Agent Switcher** from the **Tools** menu, you should see the two user agents you've just defined. You can make Firefox masquerade as either of these by selecting it from the menu.

The Sony Ericsson P910 is a mid-sized smart phone, while the Nokia 6300 is a typical mobile handset. Both are fairly old devices, but have characteristics that are typical of the class of device they represent.

Resizing the Window

When you are viewing a site destined for a mobile device, it's usually necessary to resize the browser window to approximately the right width and height to get a proper impression of how it will appear. The sites described in this guide tend to avoid absolute dimensions to

allow a wider range of devices to be supported from a single mobile layout and theme. If you view the mobile version of some pages using a typical PC sized browser window, you won't get a good impression of the appearance on a mobile device.

Using Other Kinds of Device

The sites described in this guide have been set up to work with devices in the smart phone and mobile handset categories. You can of course try accessing the site with different user agent settings, device emulators or even real devices. However, your device needs to be in the device repository to be recognized. Also, it needs to be in the smart phone or mobile handset categories of device for the site to have the right materials to generate pages. You may find that the sites fail if this is not the case.

Of course, once you've read and understood the material in this guide, you'll be able to customise the sites to add support for additional devices if you wish.

Changing User Agents in Firefox

Switching user agents does *not* invalidate pages that Firefox has already cached. The consequence of this is that if you visit a page in Firefox., change the user agent and visit the page again, it's very likely that you'll simply see the previously loaded version of the page, even if you force a reload. You need to clear the cache to view the page using a different user agent. Normally you also need to restart Firefox.

You can clear the cache by choosing **Clear Private Data** from the **Tools** menu and ensuring that the **Cache** is checked. Make sure that you are not going to clear other data that you wish to keep, and then click **Clear Private Data Now**.

A good way to ensure that you see the correct version of a site is to use the following sequence:

- Start Firefox
- Clear the cache
- Select the appropriate user agent
- Access the pages you want to view

If you are using one user agent setting to view a site and you want to view it using a different one, first stop Firefox. Ensure that *all* Firefox browser windows are closed. Then repeat the sequence for the new user agent. Simply clearing the cache and selecting the new user agent normally will not work.

The user agent is not remembered across Firefox restarts.

The Jive Site

Overview

The Jive site demonstrates a range of capabilities of the Volantis Mobility Server™. It also illustrates use of XDIME 2 markup.

In particular, the site illustrates:

- XDIME 2 markup, including the use of XForms
- Laying material out differently on different devices
- Styling using themes
- Selection of different forms of content
- Templates
- Inclusion of materials from other sources.

After you've studied the Jive site, you will have a good, basic grasp of the key concepts used in the Volantis Mobility Server and of a range of techniques that can be used to construct sites.

A Quick Tour of the Site

Let's take a quick look at the site before investigating how it works. Assuming that you have successfully completed the steps described in *Installation*, on page 4, you should be able to view the site.

Start up your application server. We'll assume that it is running locally on your machine and that it is listening on port 8080. If your configuration is different, just make the appropriate changes to the URIs that you use to access the site.

You can access the Jive home page at

`http://localhost:8080/mcs/projects/jive/jivehome.xdime`.

Figure 11, on page 24, shows how the page looks when viewed using a PC browser, such as Firefox. It also shows how it looks on a mobile device. See *Simulating Mobile Device Access* on page 5, for a simple way to get an idea of how pages will look on mobile devices.

In particular, the home page is split into 3 separate, interlinked pages when accessed from a mobile device. This is one way of making large pages, such as the Jive home page, available to small devices. We'll see exactly how this is achieved later.

Most links in the site are active. For example, if you click on the Management Team link in the menu, you will navigate to the page shown in Figure 3 on page 10.

Some links, such as those to the CNN news articles and the Advanced Search entry in the main menu are actually inactive. They simply route back to the page in which they occur.

As you navigate around the site, notice that there are substantial differences in the way the site appears on the PC and mobile device displays. For example, the banners on pages are different. Also, the PC version has a main menu on every page, whereas the mobile version has it only on the home page.

Despite such significant differences, the content for each page is written only once. The differences between the PC and mobile versions are achieved through adaptation based on layout, styling and media policies that you define as you create a site. We'll see how this works in detail as we look at specific pages in the Jive site.

The Pages

The rest of this section describes how the pages in the Jive site work. Pages are discussed in increasing level of the complexity of the VMS features that they illustrate.

The Management Team Page

We'll start our more detailed look at the Jive site with the management team page. This is one of the simplest pages in the site.

You can access this page either by selecting Management Team from the main menu, or by navigating to `http://localhost:8080/mcs/projects/jive/managementTeam.xdime`.

Figure 3: The management team page of the demonstration on PC and smart phone

The upper section of Figure 3 shows the management team page displayed in a PC browser. The lower section shows it as displayed on a smart phone.

Resources

This page uses the following materials:

Resource	Eclipse Navigator Location
Markup	/jive/managementTeam.xdime /jive/jiveheader.xdinc /jive/jivefooter.xdinc
Layout	/jive/WebContent/mcs-policies/jivearticle.mlyt
Theme	/jive/WebContent/mcs-policies/jive.mthm

Markup

In Eclipse, open the markup file

`/jive/managementTeam.xdime`.

If you've ever written HTML, you'll probably find a lot of this markup quite familiar. For example, look at the body of the article itself. The element `<div id="article">` is the container for the article. It contains a number of other `<div>` and `<p>` elements that contain the content itself. It also contains additional headings, in **`<h3>`** and `<h4>` elements.

Most of the markup in this file is actually XHTML 2 [3], the newest version of XHTML. Because this is an XML compliant version of HTML, it is necessary to nest elements properly and to ensure that you close elements correctly.

The Namespaces

XDIME is actually a combination of XHTML 2 with additional markup modules, which we'll look at in more detail later. It uses namespaces to allow markup from different specifications to be mixed without ambiguity. Elements from specifications other than XHTML 2 carry prefixes. For example, the the `<urid:fetch>` element is from one of the additional modules of markup provided by XDIME.

The attributes that start `xmlns`, on the top level `<html>` element, provide definitions that identify the different prefixes, such as `urid`, used within the markup. A full list of the prefixes that we normally use and the namespaces to which they refer is given in *Appendix A: Namespace Definitions*, on page 39.

Including Material

The `<urid:fetch>` element is used, in this page, to include common material from separate files. For example, the header and footer of each page is common. The `href` attribute identifies the file to be included. The header markup is contained within the file `jiveheader.xdinc` and the footer markup within `jivefooter.xdinc`. These files are in the same directory as the page itself.

The content of an included file replaces the `<urid:fetch>` element that causes it to be included.

If you are keen to see what is in the header, it is described in *The Header Include File*, on page 28. However, it does use some techniques that we have not yet covered. The footer is described in *The Footer Include File*, on page 30. It is simpler, but also uses techniques we've not yet covered.

Layout

Within the <head> element is a <link> element that refers to

`/jive/WebContent/mcs-policies/jivearticle.mlyt`

and specifies the relationship as `mcs:layout`. This is how XDIME 2 files link to the layout that they use.

Layouts define the physical position of markup. Different versions of a layout can be defined for different devices or device families. These versions are known as **variants**. The effect of different layout variants is that content can be repositioned to be most appropriate for use on a particular device without having itself having to be changed.

To achieve variation in layout, markup is targeted at particular, named areas within the layout. These areas are known as **panes**. Panes are defined as rectangular areas within a

layout. Markup is associated with a particular pane, and appears there once VMS has processed the page.

You can see how this works by opening the layout

`/jive/WebContent/mcs-policies/jivearticle.mlyt`

in Eclipse. Select the `Variant` labelled `Targeted, Mobile` and click on the `Design` tab.

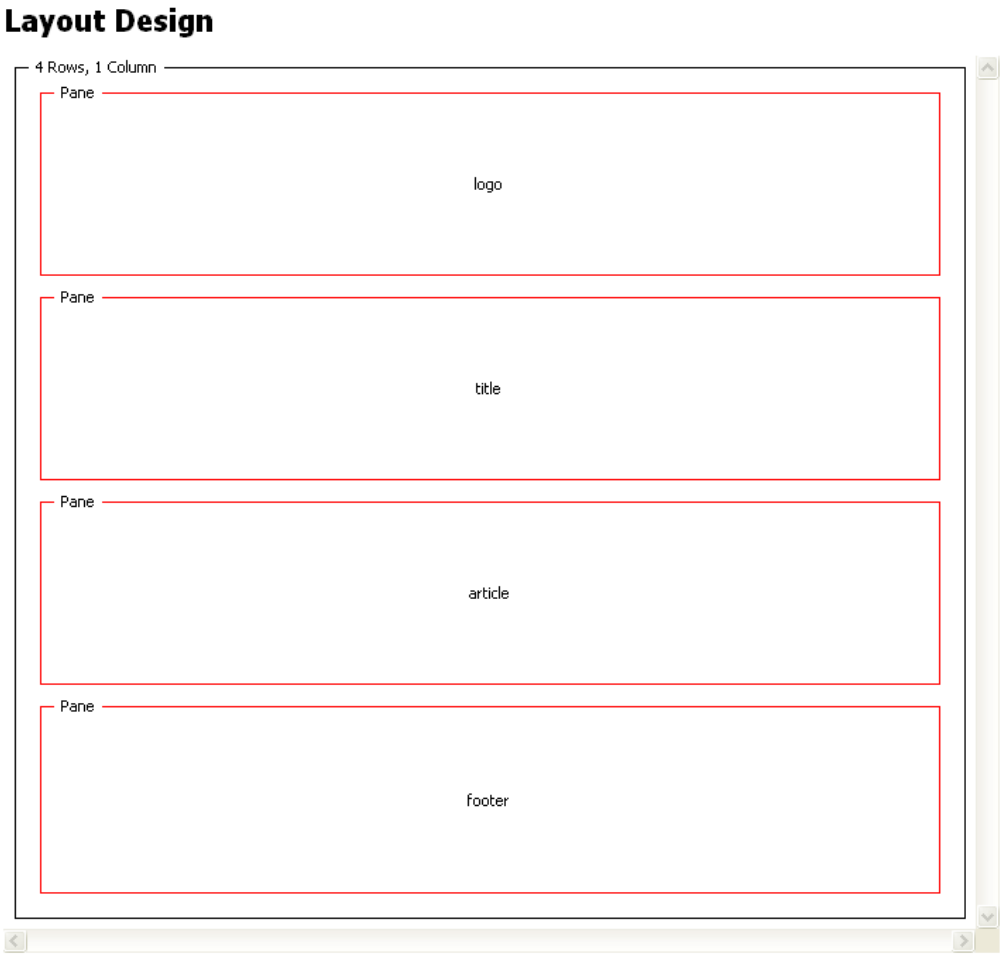

Figure 4: Article layout for a mobile device

Figure 4 shows the version of the layout for the mobile device. It includes 4 panes named `logo`, `title`, `article` and `footer`. The markup in `/jive/managementTeam.xdime` directly provides the content for the `title` and `article` panes. The included files provide the content targeted to the `logo` and `footer` panes.

The PC variant of the layout contains these same 4 panes but has others in addition. This particular markup makes use of the theme to associate markup with panes. We'll see how this is achieved in *Targeting Content to Panes using Themes*, on page 13.

Styling

Within the <head> element is a <link> element that refers to

`/jive/WebContent/mcs-policies/jive.mthm`

and specifies the relationship as `mcs:theme`. This is how XDIME 2 files link to the themes that they use.

Themes define the styling associated with markup. Different versions of a theme can be defined for different devices or device families. These versions are known as **variants**. The effect of different theme variants is that content can be styled to be most appropriate for use on a particular device without having itself having to be changed.

You can think of a theme variant as a being very similar to a CSS stylesheet. Indeed, VMS uses the standard CSS names for style properties and selectors. A theme is really a collection of stylesheets. Which stylesheet is used depends on the particular device accessing the page.

Open the theme for this page in Eclipse. Select the `Variant` named `Targeted, PC` and click on the `Design` tab. In `Selectors`, choose `body` and in `Style Properties` select `Font`. Note that the `Family` style property is set to `"Arial", sans-serif`. This indicates that the `Arial` font will be used for all content in the `<body>` element and a `sans-serif` font if `Arial` is unavailable. In addition, the `Size` property is set to `small`. Of course, the values of these properties could be different in a variant of the theme for a different device. Indeed, though the same fonts are specified in the mobile variant, the `Size` property is set to `x-small`. You can check this by clicking on the `Overview` tab, then selecting the mobile variant and choosing the `body` selector and the `Font Style Properties`.

The vast majority of Style Properties that can be set in a theme correspond directly with those in CSS 2. A small number of additional properties are available to control features that are specific to VMS. The means to target content to panes is an example of such an extension.

The styling used on this page is basically the same as that for the home page. Styles are used to achieve the banner effects in the headings and to define the outline of the individual subsections. More details about the styling are given in *Content Styling*, on page 25.

Targeting Content to Panes using Themes

VMS provides a number of mechanisms for targeting content to panes. Different mechanisms are more convenient in different circumstances. In the markup for articles, such as the management team page, the mechanism used relies on the ID attributes of specific `<div>` elements, together with entries in the theme.

You can see how this works by examining the `<div>` for the article itself. The ID attribute of this `<div>` attribute is `article`. The theme contains an entry for the element with this ID. In Eclipse, open the PC variant of the theme at

`/jive/WebContent/mcs-policies/jive.mthm`

in `Design View`. Choose the selector `#article`. This selector identifies those properties associated with markup elements whose ID is `article`. In `Style Properties`, select `Layout` to see the layout-related properties for this selector. The `Target Container` property is set to `article`, the name of the pane in which the content of the article is to appear. Of course, there is no particular reason for the pane name and the ID attribute to have the same value, though in this case it makes things nicely consistent.

If you have a running system, you can see the effect that this entry has. Delete the value of the `Target Container` property, save the theme and reload the page in a PC browser. The article itself does not appear, though the rest of the page does. Reinstate the value `article` in the `Target Container` property, save the theme again and reload the page. The article text reappears.

Themes can be used to associate panes with content using any selectors, not just those based on ID attributes. We'll see alternative approaches, based on use of styles in-line in the markup later in this guide.

It's worth remembering that content will not appear unless it is associated with a pane in the layout.

Header and Footer

The header and footer of the page are defined in the files that are included by `<urid:fetch>`. The way in which the different headers are created is described in *The Header Include File*, on page 28. Similarly, creation of the footer is described in *The Footer Include File*, on page 30.

Content Inclusion

We've already seen that this page makes use of content inclusion for its header and footer. More details about content inclusion are given in *Content Inclusion* on page 28.

The Human Resources Page

Figure 5: The human resources page of the demonstration on PC and smart phone

The upper section of Figure 5 shows the human resources page displayed in a PC browser. The lower section shows it as displayed on a smart phone.

Resources

This page uses the following materials:

Resource	File
Markup	/jive/humanResources.xdime /jive/jiveheader.xdinc /jive/jivefooter.xdinc
Layout	/jive/WebContent/mcs-policies/jivearticle.mlyt
Theme	/jive/WebContent/mcs-policies/jive.mthm

What it Shows

This page shows exactly the same capabilities as described in *The Management Team Page*, on page 10.

Linked Pages

In addition to the menu links, the link to **Current open positions** within the **Job Opportunities** section is active.

The Calendar Page

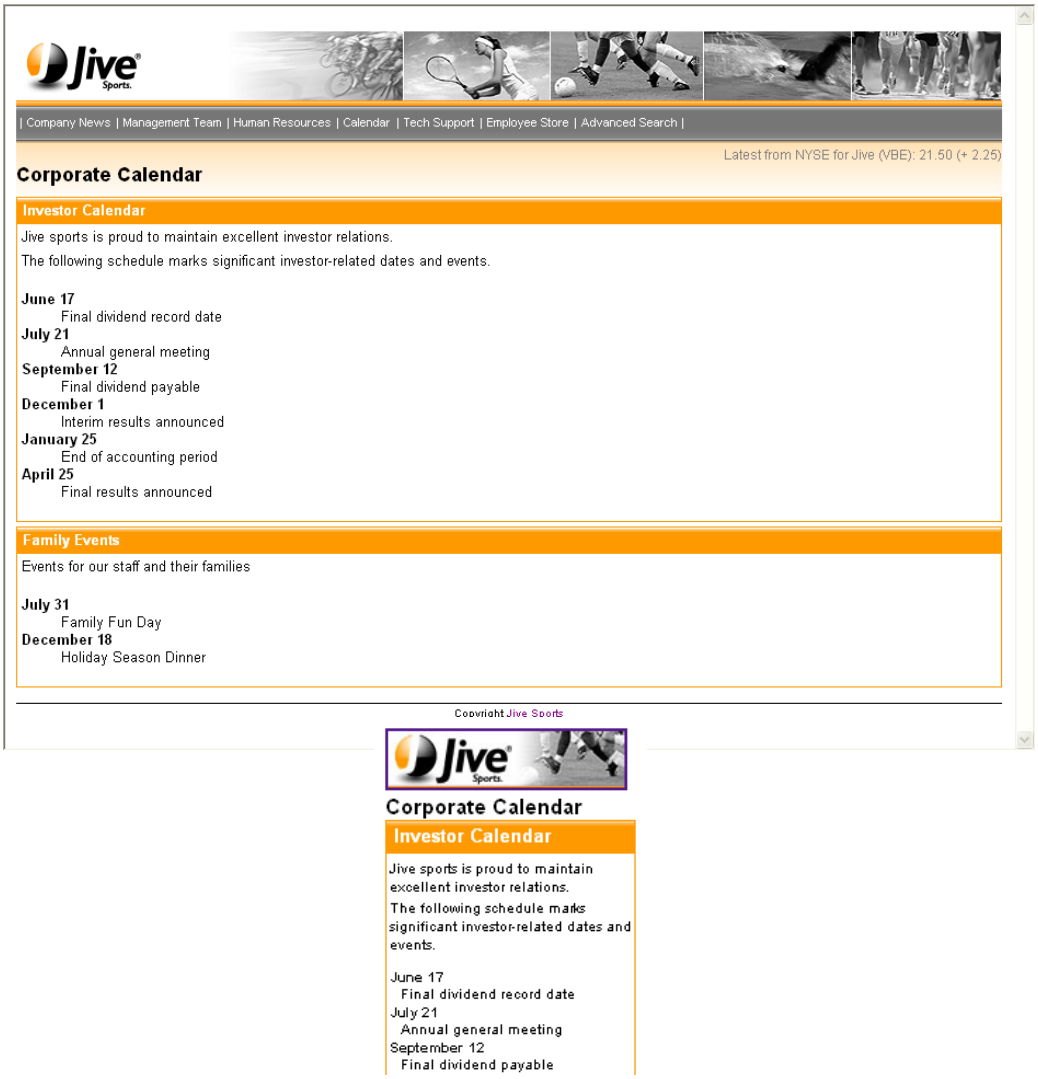

Figure 6: The calendar page of the demonstration on PC and smart phone

The upper section of Figure 6 shows the calendar page displayed in a PC browser. The lower section shows it as displayed on a smart phone.

Resource	File
Markup	/jive/calendar.xdime /jive/jiveheader.xdinc /jive/jivefooter.xdinc
Layout	/jive/WebContent/mcs-policies/jivearticle.mlyt
Theme	/jive/WebContent/mcs-policies/jive.mthm

What it Shows

This page shows basically the same capabilities as described in *The Management Team Page*, on page 10. Some different markup is used. In particular, the individual calendar entries are specified as entries in definition lists.

The Technical Support Page

Figure 7: The technical support page of the demonstration on PC and smart phone

The upper section of Figure 7 shows the technical support page displayed in a PC browser. The lower section shows it as displayed on a smart phone.

Resource	File
Markup	/jive/techSupport.xdime /jive/jiveheader.xdinc /jive/jivefooter.xdinc
Layout	/jive/WebContent/mcs-policies/jivearticle.mlyt
Theme	/jive/WebContent/mcs-policies/jive.mthm

What it Shows

This page shows exactly the same capabilities as described in *The Management Team Page*, on page 10.

Mail Links

The page includes links of the type `mailto:` which will cause mail applications to start on some phones when selected. The e-mail addresses are fictional.

The Company News Page

Figure 8: The company news page of the demonstration on PC and smart phone

The upper section of Figure 8 shows the company news page displayed in a PC browser. The lower section shows it as displayed on a smart phone.

Resource	File
Markup	/jive/companyNews.xdime /jive/jiveheader.xdinc /jive/jivefooter.xdinc
Layout	/jive/WebContent/mcs-policies/jivearticle.mlyt
Theme	/jive/WebContent/mcs-policies/jive.mthm

This page shows basically the same capabilities as described in *The Management Team Page*, on page 10. However, it also illustrates another way in which content can be selected for use on different devices.

Content Selection Using Markup

This page includes some titles that are deliberately too long to be used comfortably on a small mobile device. You can see them in the <h3> and <h4> elements in the markup. The `<sel:select>` element and it's children, `<sel:when>` and `<sel:otherwise>` control which version of the content appears. These elements are an implementation of the DISelect content selection mechanism that is part of the W3C Device Independent Authoring Language [1]. Details of the mechanism are given in the companion W3C specification Content Selection for Device Independence [2].

The effect of the elements is cause different text to be used for the headings in the PC and mobile versions of the page. The expression in the `expr` attribute of the `<when>` elements

controls whether or not the markup they contain is expressed in the page. For example, the expression

```
device:getAncestorRelationship('PC') != 'ancestor'
```

evaluates to `false` if the page is to be rendered on a PC device and `true` otherwise. The function `device:getAncestorRelationship()` is documented in the Multichannel Server help within Eclipse. Search for the function name using a Search Scope of MCS to find the information. Other functions that can be used in these expressions are documented in the help under the topic MCS function expressions.

So, for example, in the first news item, the text `Jive at the Olympics` is used for the heading on the mobile version and the text `New York - July 20: Jive selected as supplier to Team USA Basketball Olympic squad` is used for the PC version.

Note that one additional namespace,

`http://www.w3.org/2004/06/diselect`

is specified in the root element of the page to allow the selection markup to be used with the prefix `sel`.

Content Styling

The styling used on this page is the same as that for the home page. That is described in more detail in *Content Styling*, on page 25.

The Employee Store Page

Figure 9: The employee store page of the demonstration on PC and smart phone

The upper section of Figure 9 shows the employee store page displayed in a PC browser. The lower section shows it as displayed on a smart phone.

Resource	File
Markup	/jive/employeeStore.xdime /jive/jiveheader.xdinc /jive/jivefooter.xdinc
Layout	/jive/WebContent/mcs-policies/jivearticle.mlyt
Theme	/jive/WebContent/mcs-policies/jive.mthm

What it Shows

This page shows many of the the same capabilities as described in *The Management Team Page*, on page 10. In addition it illustrates the use of a form.

Using XForms for Forms

The page illustrates the use of an XForm form that allows users to check the availability of items for purchase. XForms is the forms markup within the W3C DIAL specification[[1]]. You can see the markup for the form in the page itself. There are two parts.

The Form Data Model

Within the <head> element in the page, there is an <xf:model> element that defines the data model for the form. In particular, within this element is the <xf:submission> element that specifies the URI to be used when submitting the form.

The Form Controls

The form controls appear within the body of the page. Each of the two <xf:select1> elements defines a single selection list. One of these contains a list of products, the other a list of sizes. Each individual item in a list is defined by an <xf:item> element. Each item contains a label and a value.

The <xf:submit> element defines the control by which the form is submitted. In this case it appears as a button labelled Check. The element includes a submission attribute that references the <xf:submission> element within the model. This in turn defines the URI to be used during submission.

Processing

The form data is processed by a simple JSP page, **checkAvail.jsp**. It is described in *The Availability Page*, on page 22.

The Availability Page

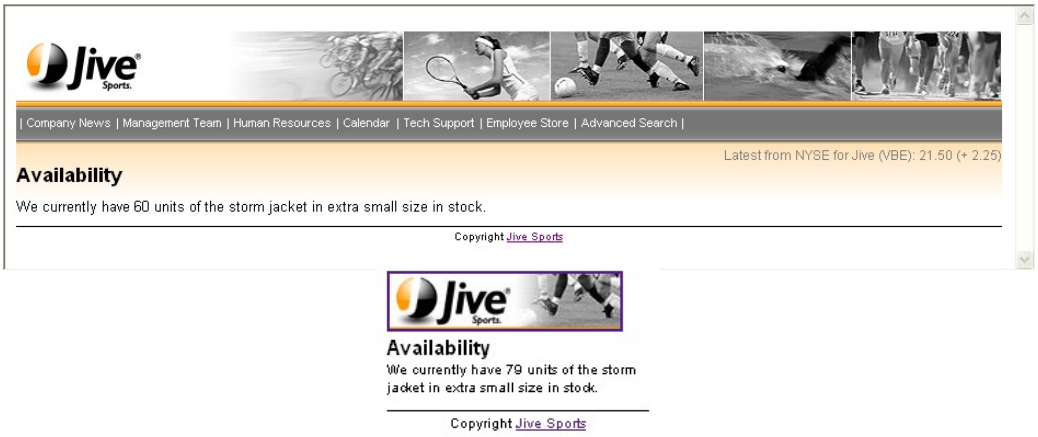

Figure 10: The availability page of the demonstration on PC and smart phone

The upper section of Figure 10 shows the availability page displayed in a PC browser. The lower section shows it as displayed on a smart phone.

Resource	File
Markup	/jive/checkAvail.jsp /jive/jiveheader.xdinc /jive/jivefooter.xdinc
Layout	/jive/WebContent/mcs-policies/jivearticle.mlyt
Theme	/jive/WebContent/mcs-policies/jive.mthm

What it Shows

This page shows exactly the same capabilities as described in *The Management Team Page*, on page 10. However, it also illustrates how XDIME can be generated dynamically by a Java Server Page (JSP).

Using JSP Pages to Generate XDIME 2

The page illustrates how to generate XDIME 2 from a JSP page. You can see the source in `/jive/checkAvail.jsp`. The processing simply calculates a random number and inserts it into the markup at the appropriate points.

Before we look at the code in the page, it is worth noting that the XML statement

```
<?xml version="1.0" encoding="UTF-8"?>
```

MUST appear as part of the first line of the file. This is because JSP pages tend to generate a lot of white space during processing. The XML statement MUST appear as the first line of the output from the page to comply with XML rules. If it doesn't the resulting XDIME 2 markup will be invalid and an exception will be thrown.

The page import definition that also appears on the first line of the file makes the java mathematical functions available to the JSP.

The first block of code (enclosed in the <% and %> tags) initializes the result and then sets the content type. For XDIME pages the content type MUST be set to

```
x-application/vnd.xdime+xml
```

This content type is recognized by VMS and causes adaptation to be performed. Other content types are passed through unchanged by VMS.

The parameters from the request are retrieved into the java variables `size` and `item`. The values of the parameters are the contents of the `<xf:value>` element associated with the item selected by the user before submitting the form.

The value for the available stock is computed as a random number and rounded to an integer in java variable `stock`.

After the initial block of code, the XDIME markup for the majority of the page appears. This is very similar to the other pages we have already investigated. The differences are near the end of the page where the values of the java variables `stock`, `item` and `size` are used to construct the message returned to the user.

There is nothing special about the use of JSP facilities in this page. The only difference is that XDIME is being generated rather than, for example, HTML.

The Home Page

Figure 11: The home page of the demonstration on PC and smart phone

The upper section of Figure 11 shows the home page displayed in a PC browser. The lower section shows it as displayed on a smart phone. On a mobile device the home page is actually converted to three, separate smaller pages with appropriate navigation. This process of splitting a large page into a number of smaller pages is known as fragmentation. It is described in *Different Content Structure*, on page 25.

Resource	File
Markup	/jive/jivehome.xdime /jive/jiveheader.xdinc /jive/jivefooter.xdinc
Layout	/jive/WebContent/mcs-policies/jivehome.mlyt
Theme	/jive/WebContent/mcs-policies/jive.mthm

What it Shows

Different Banner

The PC and smart phone versions of the page banner are different. This is achieved through a combination of layout and image variants. These are specified via the header file, `/jive/jiveheader.xdinc`.

The section *The Banner*, on page 29, gives details of how the banner is defined and rendered..

Different Menu Styles

The PC and smart phone versions of the main menu are entirely different. This is achieved through a combination of layout and image variants. Again these are specified via the header file, `/jive/jiveheader.xdinc`. The section *The Main Menu* on page 29 gives details of how this is achieved.

Different Content Structure

When rendered on a PC, all the material is presented as a single page. When rendered on mobile devices, including smarth phones, the material is presented as a series of three interlinked pages. Material in the left column of the PC version of the page is reached via the `Company Information` link in the mobile version. Material in the right column of the PC version of the page is reached using the `News Feeds` link in the mobile version.

This structural difference is achieved using a technique called *Fragmentation*. This is based on the layout for the page. You can see it being used in the mobile variant of the layout. Open the `/jive/WebContent/mcs-policies/jivehome.mlyt` layout and select design view on the mobile variant. Notice the yellow wire frames labelled `Fragment`. These represent pages that will be presented to the end user. Each fragment contains one or more panes. Content directed to these panes will appear in the fragment that contains them. Notice that this completely decouples the structure of the pages from the structure of the content allowing complete freedom in the way that material is fragmented on different devices without affecting the content itself.

Properties associated with the fragment define the text used in the various links used to navigate between them and can be used, together with the theme, to control the styling applied. For example, select the fragment named `companyInformation` in the layout and look at the property `Link Text`. The value is the string `Company Information`. This is the text that appears on the links that allow navigation to this page.

Content is targeted to panes using the theme, as described in *Targeting Content to Panes using Themes*, on page 13.

Content Suppression using Layout

The banner illustrates content suppression using layout. See *Content Suppression in the Banner*, on page 29 for details.

The main menu illustrates content suppression using layout. See *Content Suppression in the Main Menu*, on page 30 for details.

Content Styling

Sections within the page have colored borders and use background images to style headings. In the content, the `<div>` and `<h3>` elements that have this styling have style classes specified. You can see the style properties by viewing the theme

`/jive/WebContent/mcs-policies/jivehome.mthm`.

Open the PC variant in design view.

The style classes `company_item` and `external_item` control the styling of the `<div>` elements. Choose the style selector `div.company_item` and expand the style property `Border`. Select property `Border Color` and note that it is set to a shade of orange (#f90) for all borders. `Border Style` is set to `solid` and `Border Width` to 1 pixel. These are standard CSS properties.

The style classes `company_header` and `external_header` are used to control the styling of the `<h3>` elements. Choose the style selector `.company_header` and select the style property `Background`. Notice that the `Image` property is set to `/images/orange_header.mimg` and the `Repeat` property is set to `repeat-x`. Again, these are standard CSS style properties that control the background image for the element to which they are applied. The image is 1 pixel wide and is replicated horizontally by the browser to form the background.

Similar definitions are used in the mobile version of the theme, with similar results.

Content Inclusion

The page illustrates the use of content inclusion for its header and footer. See *Content Inclusion* on page 28 for details.

Linked Pages

In addition to the menu links, the links within the **Announcements** and **Jive Human Resources** sections are active.

The Advanced Search Page

This page is not active in this version of the demonstration.

In terms of how it is rendered, it shows exactly the same capabilities as described in *The Management Team Page*, on page 10.

Content Inclusion

The content for the headers and footers is contained in separate files that are included by each page.

The header include file, described in *The Header Include File* on page 28, provides the banner and main menu. The footer include file, described in *The Footer Include File* on page 30, provides the copyright notice and a link back to the home page.

Mechanism

Content is included using VMS's Dynamic Content Integration (DCI). In this particular case, no processing of included material is necessary. The include files are written in XDIME 2 and can be used without any transformation. Consequently, the simplest XDIME 2 `fetch` mechanism can be used.

You can see the markup that controls the inclusion in the home page content file

`/jive/jivehome.xdime`.

The statement `<urid:fetch href="jiveheader.xdinc"/>` causes the material from the header file to be included. In this case, the `href` attribute specifies the relative URI of the material to be included.

Later in the file, the footer is included with the statement `<urid:fetch href="jivefooter.xdinc"/>`

The result of inclusion is that the content of the referenced file replaces the statement that causes it to be included.

This mechanism for inclusion is one of the extensions that XDIME provides over and above the capabilities of the W3C Device Independent Authoring Language [1].

Consequences

Since common headers and footers are used throughout the demonstration, changes can be made in one place, namely the appropriate include file.

It would be possible to have exactly the same header and footer on every page of both the PC and smart phone versions of the site. In this particular demonstration, however, VMS capabilities have been used to modify the appearance of the header on different pages. This is covered in detail in *The Header Include File*, on page 28.

The Header Include File

The header include file is used in every page of the demonstration. It provides a common banner and menu for each page. The file is `/jive/jiveheader.xdinc`.

You can see the banner and main menu material in both PC and smart phone versions of the home page in Figure 11 on page 24.

Markup

The header includes markup for the page banner, the main menu and the current stock price. The stock price is simply fixed text in this demonstration. In a real system, it would be retrieved from some kind of data feed.

Panes are assigned using in-line styles. For example, the first `<div>` element in the banner has a `style` attribute that sets the `mcs-container` style property to `logo`. This has the effect of targeting the content of the `<div>` to the pane named `logo`. Note that the value `logo` is a string and hence must be quoted within the style attribute.

The other `<div>` elements use the same technique to target their content to the appropriate panes.

Each `<object>` element in the markup references an image component. Images are shown if suitable variants are available for the particular device being used. If there is no suitable image, the material in the body of the `<object>` element is used instead. In this markup, each `<object>` element contains text that can be used instead of the image.

You can see the effect if you access the site using the methods described in *Simulating Mobile Device Access*, on page 5.

In addition to using an in-line style to associate the stock price with a particular pane in the layout, it is also used to set the color of the text. In-line styles can be used to define style properties. Just about any CSS 2 property can be specified this way. When the same style property is to be used irrespective of the device accessing a page, in-line styles can be effective. As an alternative to using the style attribute on individual elements, it is also possible to specify styles using a `<style>` element within the `<head>` element.

Themes are particularly useful for defining styles that differ between devices. Often fonts, colors, margins and padding need to be different on different devices. In such cases themes provide a simple way to control device-dependent styling.

The Banner

The PC and smart phone versions of the banner are quite different. The PC version uses a pair of images, one for the Jive logo and a separate one for the sports-related imagery. The mobile version uses a single, smaller, image that combines logo and imagery. A combination of layout and image variants allows the versions to use the different images.

Content Suppression in the Banner

The main banner image is defined by the image component at

`/jive/WebContent/mcs-policies/images/jive_logo.mimg`

The PC variant of this image contains just the logo, while the mobile variant includes additional sports imagery.

The separate sports imagery used in the PC version is defined in the component at

`/jive/WebContent/mcs-policies/images/photos.mimg`

This image is not used in the mobile version of the site.

The markup, in `/jive/jiveheader.xdinc`, is the same for both PC and mobile versions. You can see that both image components are referenced from `<object>` elements in the banner section of the markup. To prevent the sports imagery from appearing in the mobile version, we arrange to omit the pane to which it is targeted.

The pane used for the logo is named `logo` and the pane used for the additional imagery pane is named `photos`. In the mobile variant, only the `logo` pane is present. You can see this by browsing the layout at `/jive/WebContent/mcs-policies/jivehome.mlyt` in Eclipse. The PC variant includes a pane named `photos`, while the mobile variant does not.

If a pane does not appear in a layout, content targeted at that pane is suppressed. This mechanism provides a very simple form of content selection.

The Main Menu

The PC and smart phone versions of the main menu are entirely different. This is achieved through a combination of layout and image variants.

Different Representations of Menu Items

Each menu item is an image component. For example

`/jive/WebContent/mcs-policies/menu_calendar.mimg`

is the component that represents the variants used for the calendar menu item. The PC version is an image of the text representing the menu item. The smart phone version is an image representing a suitable icon. You can see the images in Eclipse. They are in the directory `jive/images`. PC and smartphone variants are in their own directories below this. Double click an image file to view it.

Smartphones are a subset of mobile devices. If this particular site is accessed by a mobile device that is not a smartphone, text will be shown instead of the images.

Different Layout

The layout of the menus differs between the PC and smart phone versions. Each menu item is assigned to a particular pane. For example, we can see from the markup that the `calendar` menu item is assigned to the pane `menu_calendar`. In the PC version of the layout, the panes are arranged in a single row. In the smart phone version they are arranged on a two row, three column grid. You can see this by browsing the layout at

/jive/WebContent/mcs-policies/jivehome.mlyt

This is the layout used for the home page. Open Design View on the PC variant. You can see the row of panes for the menu items. In contrast, in the mobile variant, the menu item panes appear in a 3 x 2 grid.

Content Suppression in the Main Menu

Menu Item Suppression

The menu item Advanced Search appears only in the PC version of the main menu. As with the banner, the content is suppressed in the mobile version. The pane menu_advanced_search, to which this menu item is targeted, does not appear in the mobile variant of the layout.

Suppression of the Entire Menu

The same approach is used to suppress the entire menu in the mobile version of pages other than the home page. None of the panes to which the menu items are targeted appear in the mobile variant of the layout at

/jive/WebContent/mcs-policies/jivearticle.mlyt

Background Images in the Main Menu

In the PC version, the main menu uses an image to provide its background. The image component

/jive/WebContent/mcs-policies/images/repeater_menu.mimg

defines the image. It is 1 pixel wide, and is stretched automatically to create the background of the layout area that contains the panes for the individual menu items.

You can see how this is specified in

/jive/WebContent/mcs-policies/jivehome.mlyt

the layout for the home page. Open Design View on the PC version and select the 1 Row by 1 Column grid that surrounds the 1 Row by 7 Columns grid that itself contains the panes for the main menu. In Format Attributes, the Background Component references the background image /images/repeater_menu.mimg.

The Footer Include File

The footer include file is used in every page of the demonstration. It provides a common copyright notice and a link to the home page. The file is /jive/jivefooter.xdinc.

Layout

The contents of the footer are targeted to the pane named footer in the layout of the page in which the file is included. The footer illustrates another way of targeting content to a particular pane. Rather than using an in-line style, this method makes use of the theme. The element that contains the footer has an ID of footer. In both the PC and mobile variants of the theme the selector for this ID defines the pane to which its content is to be targeted.

You can see the association in the Theme Design for either variant of the theme, which is at

/jive/WebContent/mcs-policies/jive.mthm

Look in the Layout section of the style properties for selector #footer. The Target Container property is set to footer, the name of the pane. This is equivalent to setting the style attribute in the markup, the mechanism we've used elsewhere.

Using the theme, to identify the pane, in this way can be convenient, particularly if other styling is desired on the material to be targeted to a particular pane. This is the case here. We'll see the additional styling in the following section. In addition, of course, it can be used to allow different panes to be targeted by the same content on different devices. In particular this can allow material to be combined in different ways on different devices.

Styling

The copyright notice is centered using the theme. In the Theme Design for either variant, you can see that in the `Text` section of the style properties for selector `#footer`, the `Align` property is set to `center`.

Note that the enclosing `<div>` element defines the XHTML 2 namespace.

You can see the copyright notice and the link in both PC and smart phone versions of each page. For example, it appears at the bottom of the home page shown in Figure 11, on page 24.

The Device Information Page

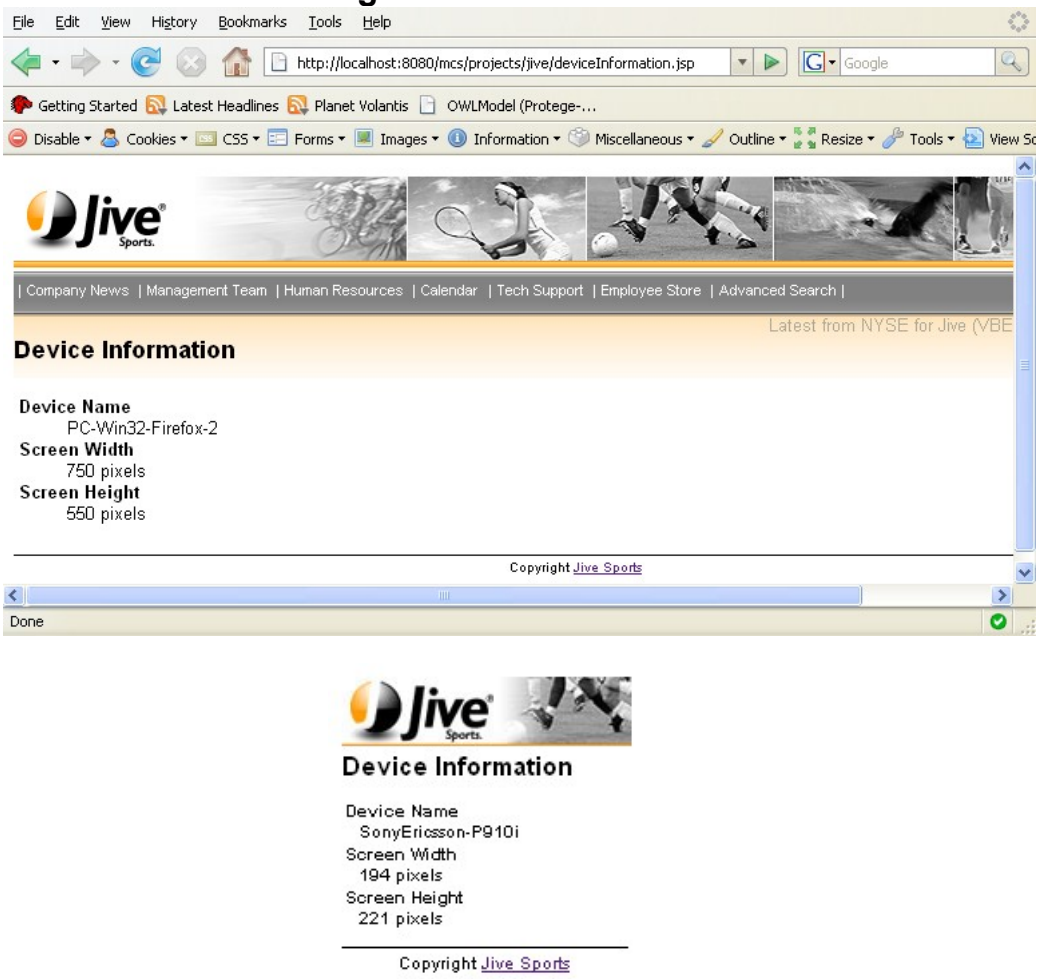

Figure 12: The Device Information Page, on PC and Smart Phone

Figure 12 shows the device information page as displayed on a PC and a smart phone.

To access this page, enter the URI

`http://localhost:8080/mcs/projects/jive/deviceInformation.jsp`

into your browser.

Resource	File
Markup	/jive/deviceInformation.jsp /jive/jiveheader.xdinc /jive/jivefooter.xdinc
Layout	/jive/WebContent/mcs-policies/jivearticle.mlyt
Theme	/jive/WebContent/mcs-policies/jive.mthm

What it Shows

This page shows basically the same capabilities as described in *The Availability Page*, on page 22. In particular, it illustrates how XDIME can be generated dynamically by a Java Server Page (JSP). However, it also shows how to use data from the device repository programmatically.

The data displayed in this page is taken from the device repository. The JSP uses calls to the VMS `device` object that is available to every page. The method `getPolicyValue()` allows code in the JSP to access information about the device currently being used to

access the page. In the case of this page, the width and height of the display are retrieved and written to the output page along with the name by which VMS knows the device.

You can access a variety of properties associated with the device. Each property has a name. For example, `pixelsx` is the width of the usable display area in pixels.

You can find these so-called short names for properties from the device repository. Open the `devices.mdpr` file in the `deviceRepository` project, and choose the **Structure** tab. Expand the `Output` item and select `Width of usable display area in pixels`. Note that at the top of the **Policy Definition** window, the `Short Name` is shown as `pixelsx`.

The Welcome Page Site

Overview

The welcome page site demonstrates a few extra capabilities of the Volantis Mobility Server™.

In particular, the site illustrates:

- The use of templates within XDIME 2
- Passing simple and complex parameters to templates

The Welcome Page

The site contains a single page. That page is constructed from a main content file, and include file and a template.

Start up your application server, if it is not already running. We'll assume that it is running locally on your machine and that it is listening on port 8080. If your configuration is different, just make the appropriate changes to the URIs that you use to access the site.

You can access the Welcome home page at

`http://localhost:8080/mcs/projects/welcome/welcome.xdime`

Figure 13, on page 36, shows how the page looks when viewed using a PC browser, such as Firefox. It also shows how it looks on a mobile device. See *Simulating Mobile Device Access* on page 5, for a simple way to get an idea of how pages will look on mobile devices.

The PC and smartphone versions differ mainly in layout.

Open the page in Eclipse. It is at `/welcome/welcome.xdime`.

Much of the material should look very familiar by now. The new elements are those carrying the `template` prefix. In XDIME, templates are included material that can be tailored by the use of parameters. The template used in this page can be found at

`/welcome/XDIMETemplates/welcometemplate.xdtpl`.

Open the template in Eclipse.

In the welcome page, the `<template:apply>` element defines the template to be used. The `href` attribute is a relative URI to the template to be used. The `<template:binding>` elements define values to be passed into the template and used within it.

The first of these elements defines the value of the `linkstitle` parameter. It is a simple text string and so is specified as the value of the `value` attribute.

The second `<template:binding>` is more complex. This time it defines some markup to be passed in and used in the template. The element contains a `<template:complexValue>` element, whose content is the markup to be passed in.

Within the template itself, the <template:declarations> element contains the declarations of the `linkstitle` and `congratulations` parameters. The first of these is declared as a `simple` parameter, and the second as a `complex` parameter. Note also that the declaration for the `congratulations` parameter contains a `<template:documentation>` element describing the parameter.

The body of the template contains XDIME markup. This markup replaces the `<template:apply>` element in the welcome page. Within the markup, the `<template:value>` elements define where the contents of the parameters `linkstitle` and `congratulations` appear. The values of the parameters replace the <template:value> elements when the template is processed.

Finally, near the bottom of the page, there is a familiar `<urid:fetch>` element that causes a copyright statement to be included.

Layout and Theme

Content is targeted to panes using the theme Target Container style property, in the way we've already seen in the Jive site. The layout for mobile devices is a single column whereas that for the PC places some panes side by side. Some details for the styling differ slightly between the two versions.

The layout is at `/welcome/WebContent/mcs-policies/welcome.mlyt`

The theme is at `/welcome/WebContent/mcs-policies/welcome.mthm`

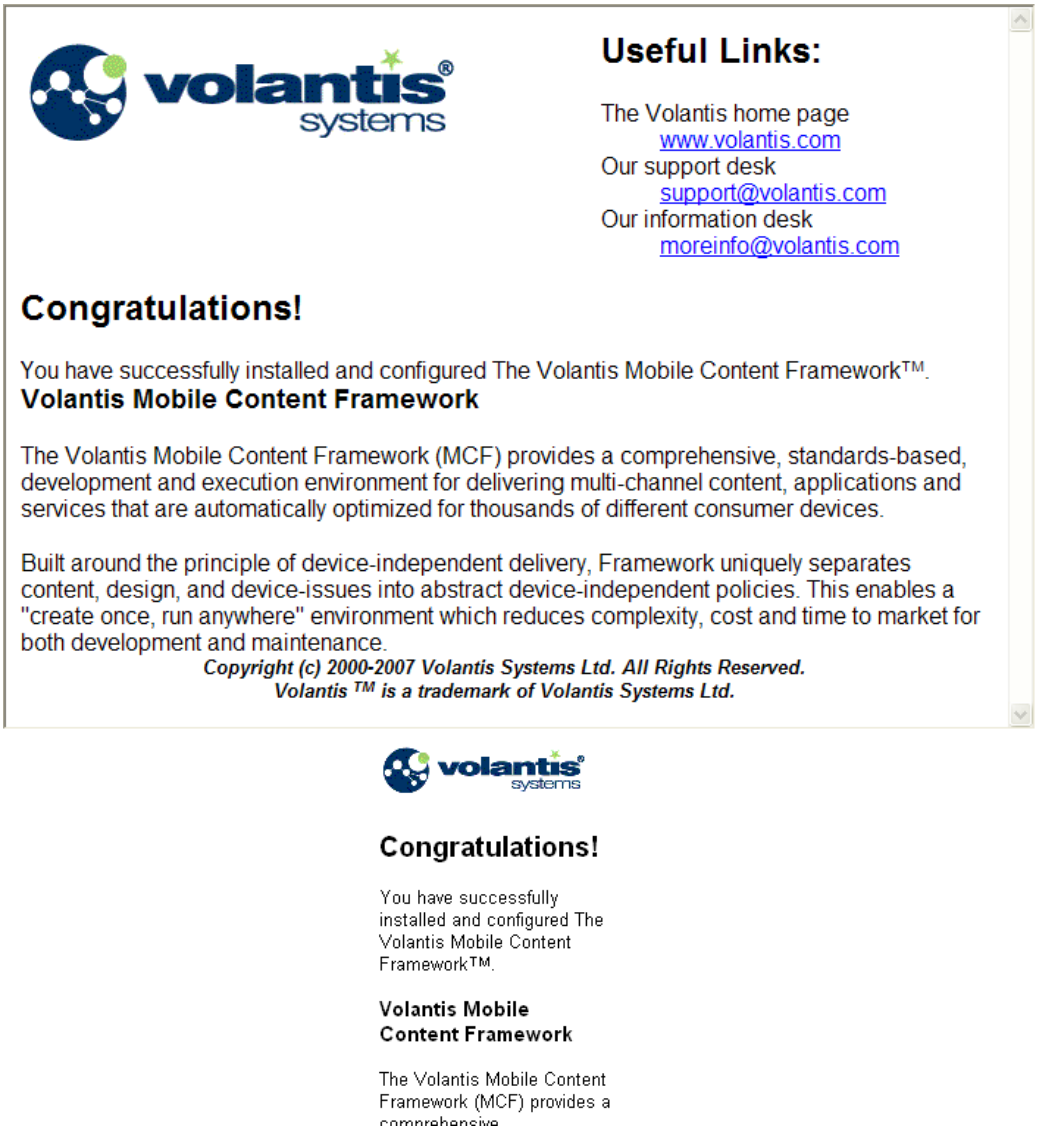

Figure 13: The Welcome page on PC and smartphone

Playing with the Pages

The sites described in this guide are set up so that changes made in markup, layout or theme should appear immediately in the running web application. Try making small modifications to the materials to see what happens.

Here are a few suggestions to get you started.

Targeting Content

Pane Names

With a PC browser, navigate to the Jive home page at

`http://localhost:8080/mcs/projects/jive/jivehome.xdime`

Open the layout `/jive/WebContent/mcs-policies/jivehome.mlyt` in Eclipse. Choose the PC variant in Design view. Click on the `logo` pane. Rename it to logo2 and save the layout.

Refresh the page in your browser. The Jive logo disappears. When content is targeted at a non-existent pane, it does not appear in the rendered page.

Because the change is in the PC variant of the layout, it affects only those devices using this particular variant.

Reverse the change in the layout and save it. The logo reappears when you refresh the page in your browser.

Pane References

Using the Theme

Open the theme `/jive/WebContent/mcs-policies/jivehome.mthm` in Eclipse. Chose the PC variant in Design view. Select `#announcements` in the Selectors and `Layout` in the Style Properties. The `Target Container` field identifies the pane for this content as `announcements`. Change the value to US_news.

Refresh the page in your browser. The *Announcements* section now appears before the *CNN Around the Nation* section. They are both in the `US_news` pane. Where multiple pieces of content are targeted at the same pane, the order in which they appear in the XDIME markup determines the order in which they are rendered.

Because the change is in the PC variant of the layout, it affects only those devices using this particular variant.

Reverse the change, refresh the page in your browser and verify that the content moves back to its original location.

Using the Markup

Open the inclusion for the header, at `/jive/jiveheader.xdinc`, in Eclipse. In the first `<div>` in the banner section, change the value of the style attribute to

```
mcs-container:'footer'
```

This targets the logo to the pane used for the page footer. Save the file and refresh the home page in your browser. The Jive logo now appears in the footer of the page. The change affects all pages, since they share the same inclusion. Navigate around the site to verify this.

Because the change is in the markup, it affects all devices. Verify this by changing the user agent in Firefox to masquerade as a Sony Ericsson P910. Don't forget to restart Firefox and clear the cache as described in *Changing User Agents in Firefox*, on page 8.

Reverse the change and verify that the logo returns to the header.

Styling

Fonts

Open the theme `/jive/WebContent/mcs-policies/jivehome.mthm` in Eclipse. Chose the PC variant in Design view. Select `body` from the Selectors, and `Font` in the Style Properties. Change the value in the `Size` field to `large`.

Refresh the page in your browser. The font size of the text within various sections of the page increases. These sections inherit their font specifications from that of the `<body>` element. Paragraphs, for example, do not carry their own font specifications. You can see this by selecting `p` from the Selectors.

The headings, however, do not change. By selecting the `company_header` class or the `external_header` class from the Selectors, you can see that their fonts are specified individually. These definitions take precedence over those inherited from `body`. The rules that govern which definitions take precedence are those of CSS 2 [5].

Changes in themes affect all pages that use them. However, they affect only the devices that use the particular variant modified.

Background Colors

Again in the theme, select `body` from the Selectors, and `Background` in the Style Properties. Click the button at the left of the Color field to bring up the color picker. Select an easily recognizable color, perhaps a lurid green. Note that the RGB value appears in the text field. Save the theme and refresh the home page in your browser. The background color changes to the one you chose. Navigate around the site and note that all pages now have a colored background. In the Jive site, all pages share the same theme.

You can verify that the change does not affect the mobile version by accessing the site while masquerading as a mobile device. See *Changing User Agents in Firefox*, on page 8 for details.

Reverse the change in the theme, by deleting the contents of the text box witin the Color field. Save the theme and refresh your browser to verify that the background color has been removed.

Background Images

Again in the theme, select `body` from the Selectors, and `Background` in the Style Properties. This time, click the `Browse` button in the `Image` field. Choose the image at `/images/jive_logo.mimg`. This is the image that provides the Jive logo in the page header. In the `Repeat` field, choose `repeat`. Save the theme and refresh your browser. The Jive logo is repeated to fill the background.

Notice that the background shows through some areas of the page but not all. For example, it does not show through the two areas containing external news items. To find out why, select `div.external_item` from the Selectors. This is the style class that applies to these external news items. Select `Background` from the Style Properties. Notice that a background color is specified for this selector. Now select `div.company_item` from the Selectors. No background color is specified. The background for this selector is transparent and so the background image shows through.

Try choosing different values for the `Repeat` field. The value `repeat-x` repeats the image horizontally. You may need to increase the width of your browser window to verify this. Unsurprisingly, the value `repeat-y` repeats the image vertically.

Reverse the changes, save the theme, refresh the page in your browser and verify that the page reverts to its original appearance.

Over to You

It's easy to experiment with markup, layout and styling. Making changes and being able to see the effects immediately works because, in these sites, the configuration disables any caching.

Feel free to experiment.

Appendix A: Namespace Definitions

The following table shows the namespaces used in XDIME 2 markup and the prefixes that we normally use to identify them.

Prefix	Namespace	Comment
(none)	http://www.w3.org/2002/06/xhtml2	XHTML Version 2 [3]
xf	http://www.w3.org/2002/xforms	XForms [4]
pipeline	http://www.volantis.com/xmlns/marlin-pipeline	Volantis pipeline
template	http://www.volantis.com/xmlns/marlin-template	Volantis template
urid	http://www.volantis.com/xmlns/marlin-uri-driver	Volantis uri driver
mcs	http://www.volantis.com/xmlns/2006/01/xdime/mcs	Volantis mcs
sel	http://www.w3.org/2004/06/diselect	DISelect [2]

Appendix B: References

[1] Device Independent Authoring Language (DIAL), K. Smith, 2007, W3C Working Draft available at http://www.w3.org/TR/dial/

[2] Content Selection for Device Independence (DISelect) 1.0. R.Lewis et. al., 2007, W3C Candidate Recommendation, available at http://www.w3.org/TR/cselection/

[3] XHTML™ 2.0, J Axelsson et. al., 2006, W3C Working Draft available at http://www.w3.org/TR/xhtml2/

[4] XForms 1.0, J.M. Boyer et.al., 2006, W3C Recommendation available at http://www.w3.org/TR/xforms/

[5] Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification, Bert Bos et. al., 2007 ,W3C Candidate Recommendation, available at http://www.w3.org/TR/CSS21/