Project Title

WebDav Client for the Palm Pilot V

FINAL REPORT

ECE 145 SENIOR PROJECT WINTER 2001

Authors: Peter Trong Ho, Dinh Phuoc Hoang

Dates: 03/16/01

Introduction

Today, the World Wide Web is so popular. It seems like everyone know about it. It has so many useful applications implemented by using HTML, XML, JavaScript… However, these applications do not allow users to edit the file information of the server. That means they only give users read only access. With the booming of business and economy, people are required to travel much more than before. The Internet seems to be a solely mean for people to access information remotely. But what if we need to access and edit the information on the server? Fortunately, we have a new technology that is called WebDAV. It gives us read and write access to the information remotely. WebDAV stands for the World Wide Web Distributed Authoring and Versioning. WebDAV is extending HTTP to provide a standard infrastructure for asynchronous collaborative authoring across the Internet. This powerful protocol will allow us to edit the file content right over the server. We do not need to modify the files in local machine, and then upload to the server. WebDAV protocol gives many significant options such as: LOCK, UNLOCK, PROPFIND, MKCOL, PUT, and Other tools: - COPY, MOVE, and DELETE. These commands are very useful for us to develop application for communication device, especially those hand-held devices such as the Palm Pilot.

Background And Related Work

1. Get to know WebDAV

First, we must know about WebDav and what is it about in order to have an idea to design the project. These websites showed us helpful information about this new protocol.

http://www.ics.uci.edu/~webdav/

http://www.webdav.org

There are very helpful source codes that help us understand the functions of a WebDav client are the HTTPClient and the DAVExplore. They can be download at the website: http://www.ece.uci.edu/~hokt/bin/7.htm

2. Tools for the project

We need to have the right tools for the projects. These tools are:

1. Palm Emulator: a Palm version that runs on windows platforms. See Development Plan for downloading and installing.

2. ROM file: this is the image file of the Palm. There are many available version: Palm III is free for download from class website. PalmV can be downloaded at the PalmOS website by sign up as a developer. The Development Plan has instructions about ROM.

3. KVM: the virtual machine for embedded device. It is created by downloading and installing the J2ME kit (java micro edition) from Sun's website. Please see Development Plan for instruction.

4. Magi server: this server is developed by Endeavor technology. We download Magi software from the site: http://magi.endeavors.org/

3. Steps to develop and test the project

a. Implement the WebDav client in java

b. Using the KVM to compile and generate the .prc file

c. Download the .prc file on the emulator and test it.

d. If every work then we will down load the file onto the real Palm and using the plug-in mobile phone to connect it to the Magi server to test.

Project Specification

1. Description

In this project we will implement the Palm as a client to access data in a server that running Magi. The Magi server in this case is either a computer at home or in the office, or another Palm, which are connected to the internet. The Palm Pilot will act as a remote control. It will connect to the server by using a plug-in mobile phone. Once it is connected to the servers, the Palm will down load file information and display them on the screen. From there we can rename, edit the file on the server. After doing all that then we can load the change back to the server or we can save it on the Palm.

The input will be the IP address of the server. The output will be the file system information of the server. To connect the server, the Palm must be online. So we can use a plug-in mobile phone to connect the Palm to the internet. Once the Palm is online, we can connect to any server using the WebDav protocol.

2. Platform

In this project we will use the Palm Pilot emulator, the Palm IIIe ROM
Since this is the emulator then we do not worry about the memory constraint because we install the emulator on the PC.

We will use the plug-in mobile phone for palm as method of communication between the Palm and the servers. The tool we will use in this project is j2me kit. This is the tool kit for small handheld device with small memory resource.

3. Interface

The user interface will run as a PalmOS application. It will have button for user to press to make a command. For example, when the user tap on the start button the program will display a text field for user input the IP address of the server.

The hardware interface is the plug-in mobile phone. We will use this phone to connect the Palm to a server. But when we implement the code and using the emulator to test it we may use the serial port.

Project Design

The project will be broken into two modules. They are user interface module and communication module.

1. User interface module

The user interface will be responsible for display all the button for user to press. For example, some of those buttons are: Connect, Exit, Next.

It also is responsible for displaying the error messages, informative message. The most important part is to display the file information of the server on the Palm screen.

2. Communication module

The communication module will be responsible for the networking technique used to connect the Palm to the Magi server. It must send the appropriate request and process the server responds.

3. Interface between the two modules

When the button is pressed it will generate an event. It depends on the functionality of the button we can set which event to associate with it. For example, the connect button will associate with the event connection to the server. So upon the Connect button is pressed, the connecting procedure will be executed. The Connect button belongs to the user-interface module. The connecting procedure belongs to the communication procedure.

4. Data structure

When the file information is downloaded from the server to the Palm, it will be stored on a data structure. The file information will be just name of files or folders, size, date. So we are thinking about using a linked list.

Development Plan

1. Development Environment

Over the past couple of years, Sun Microsystems has tailored the Java Virtual Machine for products in the consumer and embedded markets. The K virtual machine (KVM) is a Java virtual machine designed for devices that have a small-memory footprint (128K of memory). This environment is optimized for small-memory, limited-resource, and connected devices such as cellular phones, pagers, personal digital assistants (PDAs), set-top boxes, and point-of-sale (POS) terminals. The KVM was developed as part of a larger effort called Java 2 Micro Edition (J2ME^{TM}) technology to provide a scalable architecture for the development of portable applications in consumer and embedded devices. J2ME addresses the technical requirements of hand-held devices and other information appliances. J2ME provides a range of virtual machine technologies

for different processor types and memory footprints. But first, we need to set up the Palm emulator before we can install the J2ME.

2. Installing the Palm OS Emulator

The task of testing and debugging Palm applications is easier when we use an emulator. This way we develop our applications and test them on the emulator first, and if everything goes well, we can load them on our Palm device.

The Palm OS Emulator (POSE) is application software that emulates the various model s of Palm devices (such as, Palm IIIe, Palm V, and so forth). We can download POSE for free from the Palm OS Emulator site. http://www.palmos.com/dev/tech/tools/emulator/

To install POSE, unzip POSE into c:\pose. Now we should have the emulator.exe file along with other files.

3. Uploading the ROM Image

First we need to down load an ROM file from the palmos website. Just go to the site: www.palmos.com and sign up as a developer. Then we can down load the ROM.

It is a good idea to put the ROM file and the emulator file in the same directory. The first time we double click on emulator, we must set the path for the ROM file. To do this click on New -> Rom File -> Other and then set the path to the Rom file. If every thing is correct then we will get a picture of the Palm like this

Figure 1 An ROM image

4. To install the KVM:

a) Down load and unzip:

Download the Connected Limited Device Configuration (CLDC). It can be download from the web page: www.sun.com/software/communitysource/j2me/download.html.

In the current release (1.0) there are two packages that we need:

Package 1: j2me_cldc_1_0_src-winsol.zip.

This package contains the CLDC reference implementation on Win32 and Solaris platforms. It includes the source code and binary runtime of J2ME CLDC, as well as tools and documentation intended for assisting us in porting the CLDC implementation to

new devices. This package is designed to be used with a profile like the new Wireless Profile outlined by the MIDP specification. The CLDC is a reference implementation that utilizes the KVM, and is the first Sun product to contain a commercial ready-to-use implementation of the KVM.

Package 2: j2me_cldc-1_0-src-palm_overlay.zip.

This package contains a CLDC-compatible port for the Palm Connected Organizer. It must be installed on top of the first package to build a compatible CLDC implementation for the Palm. Package 2 is not meant to be used separately. Download and install Package 1 first. Once we download the two packages, we can install them. Unzip the first package into the root directory C:\. This gives us a new folder called j2me_cldc, with the following directory structure:

C:\j2me_cldc\>
-- jam
-- docs
-- build
-- tools
-- api
-- kvm
-- samples
-- bin

Unzip the second package into the new directory c:\j2me_cldc.

If we are asked to overwrite some of the files, respond with *yes*. Two important files kvm.prc and kvmutil.prc installed in the *bin* subdirectory. The next section will has more information about these files.

b) Installing the KVM on the Palm

First we need to run the emulator. To install the KVM.prc and KVMutil.prc on our palm device we right click on the ROM image of the emulator, scroll down to Install Application/Database and choose Other. Then we specify the path to the KVM and KVMulti one by one. Go to the applications directory on our Palm and check to see if KVM and KVMutil are there. We should see the Duke icon as shown in Figure 1. If we do not see them, click on another application (e.g. calculator) and then go back to the applications directory.

Figure 2

If we click on either the KVM or KVMutil icon, the KVMutil application is loaded. This application allows us to set our maximum heap size and screen output options as shown in Figure 2.

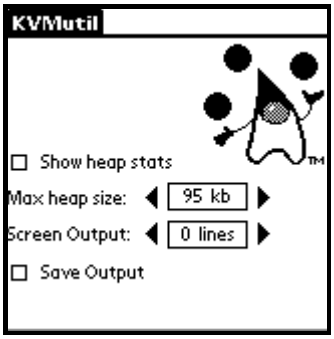

Figure 3

Now we are ready to install other applications on our Palm. There are several applications written in the Java programming language for the Palm that come with the software we just installed. All the files that end with prc in the directory c:\j2me_cldc\bin are actually Palm applications written in the Java programming language.

c) The J2ME CLDC Development Environment

When we installed the two packages, we actually installed the development environment and the documentation. The documentation is installed in the directory j2me_cldc\docs, and the classes are located in the directory j2me_cldc\bin\api\classes\database.

Another important directory is j2me_cldc\tools. The utilities in this directory can be used to generate prc files for the Palm. The exact location of the classes is j2me_cldc\tools\palm\src\palm\database.

We need must compile these classes. One way to compile those classes:

1. Go to the j2me directory
2. Enter the command on one line:

```
c:\j2me_cldc\tools\palm\src\palm\database> javac *.java
        -d c:\j2me_cldc\tools\palm\classes\database
```

   This command compiles all the classes and places them in the output directory specified by the -d option.

3. Copy the files Wrapper.prc and DefaultTiny.bmp to the new output directory

5. Platform

The platform we use to install the emulator and develop the source code is Microsoft windows. The library we need for this part is the

Com.sun.kjava.*

Javax.microedition.io.*

Java.lang.*

Kjava.* will provide us all the graphic utilities: Buttons, TextField, ScrollTextBox, paint(), getText()

Microedition.io.* will give us access to the networking utilities such as StreamConnection

Java.lang.* provides us the String class.

6. Test case

First we will test the user interface only. We will load all the buttons on the screen And output some text when they are press to prove that it work.

Second, we will test the WebDav client by connecting it to a Magi server. The Magi server and the emulator are installed on a LAN. We will test if the client can make successful connection to the server and get all the responses.

Third, we test the client by connecting to any other server such as: www.uci.edu, newport.ece.uci.edu, www.yahoo.com…. to see if it work.

Fourth, we test the database and the edit, downloading, uploading, save command of the client.

Finally, we will integrate them together and test the whole program.

7. Development schedule

We could not decide what we could do until the fifth week. Time was rushing. Those weeks before, we were trying to learn Code Warrior. But from the sixth week we must learnt Java for embedded device and we must learnt it quick since the Magi server using WebDav protocol written in Java. So the schedule for the project was:

Week 1-5: learning Code Warrior (in C)

Week 6

User interface: write code for displaying all the user buttons

Networking: write code for establishing the connection to a Magi server

Week 7:

Integrate the two modules and test the whole program as WebDav client

Present to the class what the group has done so far

Debug the program so that it work well for all testing

Week 8:

Develop a server to test the client.

Development the database to store the files or folders downloaded from the server

Figure out the request and process the responses from the server

Week 9:

Finishing the code to download files or folder from the server

Edit the files or folders

Save on the client or send them back to the server.

Week 10: In class presentation.

Final project report.

a) User interface module

First we will develop the user interface. This part is about to create all the buttons on the screen. Randy and Huu Pham are responsible for this part.

```
import com.sun.kjava.*;
import java.io.*;
import javax.microedition.io.*;

public class WebDav extends Spotlet
{
        static Graphics g = Graphics.getGraphics();
        Button connectButton;
        Button exitButton;
        TextField newText;
        ScrollTextBox scrollText;
        String addr = "", str;

        public static void main(String[] args)
        {
                (new WebDav()).register(NO_EVENT_OPTIONS);
        }

        public WebDav() // this is the constructor
        {

                newText = new TextField("Host", 10, 15, 120, 15);
                connectButton = new Button("Connect", 20, 35 );
                scrollText = new ScrollTextBox("",5, 55, 150, 85);
                exitButton = new Button("Exit",139,145);
                paint();
         }
        void paint()
        {
                g.clearScreen();
                // Draw GUI controls and buttons
                newText.paint();
                connectButton.paint();
                scrollText.paint();
                exitButton.paint();
        }
        ……..

}//end of class WebDav
```

The purpose of the inheritance of the class WebDav from class Splotlet is because Splotlet provides the event handling mechanism. It will detect and take the appropriate action when a button got pressed. In the main() function, we just create a new WebDav object. This action will call the constructor that will create each button one by one. After doing all that, it will call to the paint() function that will paint all the buttons. The key NO_EVENT_OPTIONS inside the register() means that we are using the emulator. KEY_EVENT_OPTIONS will be used instead when we do this for the real Palm device.

b) Connection module

The connection part is developed by Peter Ho and Dinh Phuoc Hoang. The most action for this part takes place inside the Connect button because this is when the program will be executed when the user press it.

```
public void penDown(int x, int y)
{
        if (exitButton.pressed(x,y))
        {
              System.exit(0);
        }
        else if (connectButton.pressed(x,y))
        {
                 if (newText.hasFocus())
                 {
                        newText.loseFocus();
                 }
                 if (newText.getText().equals("")) return;
                 addr = newText.getText();
                 scrollText.setText("Connecting to "+addr+", port 80");
                 newText.setText("");
                 scrollText.paint();
                 newText.paint();

                 try
                 {
                        StreamConnection socket = (StreamConnection)Connector.open
```

```
("socket://"+ addr + ":80", Connector.READ_WRITE,true);
if (socket == null)
{
        str = " Couldn'  t open socket";
        sc rollText.setText(str);
        scrollText.paint();
}
else
{
        String str = " Connection is established to \n MAGI on port
        80";
        scrollText.setText(str);
        scrollText.paint();

        InputStreamReader input = new
                InputStreamReader(socket.openInputStream());
        OutputStreamWriter output = new
                OutputStreamWriter(socket.openOutputStream());

        output.write("GET / HTTP/1.0\r\n");
        utput.write("Accept: image/gif, image/x-xbitmap,
                        image/jpeg, image/pjpeg, */*\r\n");
        output.write("Accept-Language: en-us\r\n");
        output.write("Accept-Encoding: gzip, deflate\r\n");
        output.write("User-Agent: Mozilla/4.0 (compatible; Palm;
                        Palm OS)\r\n");
        output.write("Host: "+ addr+ "\r\n");
        output.write(" \r\n");
        output.flush();

      // and read the response from the server
        int b = input.read(stringFromWebdav);
        if(b > 0)
        {
                String  line = new String(stringFromWebdav, 0, b);
                str = line;
                scrollText.setText(str);
                scrollText.paint();
        }
}
}//end of try

catch (Exception e)
{
        str = "Error: Unable to open connection... \n";
        scrollText.setText(str);
```

```
                    scrollText.paint();
                }
            }//end of elseif
            else
            {
                if (newText.hasFocus())
                {
                    newText.loseFocus();
                }
            }
    }//end of penDown()
```

Upon detecting the event "Connect button is press", the program will get the input IP address by using getText() and assign it to a string variable addr. Because connection is critical therefore we must put the code inside a try block. Then the program will pass the addr in the open() function. The default port used for Magi server is 80. The return type of open() function will be assign to a variable socket of type StreamConnection. After that the program will check if socket is null, it will output the "Could not open socket" message. Otherwise, it will tell that connection is established.

The next step is the send the request to the server. This will be done by the OutputStreamWriter. These requests are defaults of the server.

```
output.write("GET / HTTP/1.0\r\n");
output.write("Accept: image/gif, image/x-xbitmap,
        image/jpeg, image/pjpeg, */*\r\n");
output.write("Accept-Language: en-us\r\n");
output.write("Accept-Encoding: gzip, deflate\r\n");
output.write("User-Agent: Mozilla/4.0 (compatible; Palm;
        Palm OS)\r\n");
output.write("Host: "+ addr+ "\r\n");
output.write("\r\n");
```

After sending requests, the program will read the response from the server. This is accomplished by the InputStreamReader. For each request, the server will respond a string. The program will read each of this string and output to the screen.

8. Result

The group has successfully tested the WebDav client. We can connect to the Magi server and other server like newport.ece.uci.edu. The following is the screen shot of those tests.

Figure 3 Magi and Unix connection screen shot

PROJECT EVALUATION

1. Qualitative

The specification of the project went well during the time our group was developing it. However, the only disadvantage that we experienced was we did not have more time to do more about our project. We started with Code Warrior that was similar to C, but we ended up doing Java programming. There were a lot to find out about Java

programming for embedded device. There part that was difficult to specify was the requests and responses between our client and Magi server to down load or upload files and folders. Another difficult part was the database we tried to implement to store the downloaded data on the Palm. They certainly require us more time to find out.

The design of the group is logical. We did not have any problem with the decomposition of the project. It only took a short time for the group to figure out how to combine the two modules together. And once they are together they worked well with no conflicts at all.

The development of the project went on time with the schedule. However, because we started late we did not have more time to finish the last part of the project.

2. Quantitative

The performance of the project is good. We accomplished the first part of the project that was to design the GUI and the code to connect to a Magi server and other servers. The only difficulty was to connect to the Magi server in Endeavor Technology. It was probably because of the firewall or the protecting mechanism that the company used that was not allowed in coming requests. But we were able to connect to the Magi server we set up at home.

The other accomplishment was the implementation of the server to test the WebDav client. We successfully tested this server. It turned out that the client also worked pretty well.

CONCLUDING REMARK

Over ten weeks of researching and development we have learnt a good technique to develop a project. Analysis of the requirement of the project was very important. By doing this we will have a clear idea of what and how to do each part of the project. The division of the big project into modules and develop each module individually very efficient technique. This also makes us realize the importance of teamwork. Each group of the team was responsible for one module so that we can save more time to refine the module. In addition, we got to know about java technology for the embedded device. We all got a chance to build some useful tool for the Palm such as a user interface application. At first we did not have a clue about how the Palm may connect to the server or the internet. But now we understand about the client/server networking using Java and WebDav protocol.

The work remains for the project is to process responses from the server and to download or upload data from the server to the client and vice versa. And we also need to find out how to display the file information on the Palm screen. If we have a chance to do another project next time we will pick a topic as soon as possible to allow us more time to make it work perfectly.

CREDIT

User interface and database development:

Huu Pham, Randy Nguyen

Communication and networking development:

Peter Ho:

+ Researched and designed the source code for connecting the

Palm to the Magi server

+ Help to test and debug the program.

Dinh Phuoc Hoang:

+ Develop the source code for connecting and test it

+ Debug the program.

Credit also is give to:

Randy Nguyen: he showed us where to find the materials for the projects

and also help us debugging the program.

Kien Pham. He helped us debugging the networking code.

Clay (Project Manager) Endeavor Technology. He helped us to structure the project.

BIBLIOGRAPH

Books

Deitel & Deitel, *Java How to Program* Prentice Hall 1999

Horstmann Cay and Cornell Gary, *Core Java 2* Sun Microsystems Press 2000

Weiss Allen, *Data Structure and Problem Solving using Java*, Addison-Wesley 1999

Websites

www.webdav.org/deltav

www.webdav.org/projects

http://www.endeavors.org

http://www.endeavors.org/html/developers.html

http://www.palmos.com/dev/gettingstarted.html

http://www.ics.uci.edu/~webdav/

http://gateway.endtech.com/ccover/Magi/MagiServer/Public/org.zip

http://www.innovation.ch/java/HTTPClient/

http://www.inprise.com/jbuilder/hhe/jbhhedown.html

http://www.sun.com/software/communitysource/j2me/