네트워크 일단 가보장...

안녕하세요 핑크레드 입니다. 우아~~! 제가 점 바뻐서 늦었습니다. 무어라 변명은 언제든지..많지만..ㅋㅋㅋ. 제가 초기에 너무 한탄을 많이 해서 더 이상 해드릴 말이 있을런지 모르겠습니다. 요즘 들어서 취업도 많이 하셨는지 질문란에 새로운 질문들이 많습니다. 좋은 현상이지요. 이렇게 되면 자바 개발자들이 늘어나고 그리고 나면 많은 새로운 기술들이 나오게 될 것 같아서 너무 기쁩니다. 그래서 새로운 자바 프로그래머들을 위해서 몇 마디 드립니다. SK-VM 에 대해서 많은 분들이 잘 모르시는 것 같습니다. 현재 SK-VM 뿐만 아니라 여러 플랫폼에 대해서도 아직 부족하신 것 같습니다. 당연한 이야기겠죠..저도 초보시절에 그랬습니다. 하지만 남한테 물어보는 것은 매우 쉬운 일이죠 그렇지만 직접 찾아보는 것도 좋으리라 생각됩니다. SK-VM 의 지원 현황은 간단히 볼 수 있는 부분이 API 입니다. 그럼 무슨 API 를 봐야 하는가 MIDP API 는 모두 동일하지요 그러므로 OEM API 를 전부 살펴 보셔야 할겁니다. 그래야지 가능한 것과 불가능한 것들을 알 수가 있지요. 간단한 예를 들어 카메라가 있는 폰에서는 당연히 접근 가능한 API 가 있을 것이라고 생각됩니다. 왜냐하면 폰이 가지고 있는 기능이니깐요? 하지만 그렇지 않다는 것을 프로그래머는 알고 있어야 합니다. 왜냐하면 OEM API 에는 카메라 제어 API 가 없거든요.
이렇듯이 모든 궁금중은 OEM API 를 보면 됩니다. 차라리 사용방법을 물어보는 것이 더 하단계 진보한 질문이죠. 오늘은 짧게 잡소리 합니다. 할말이 없습니다. 언제나 말씀드리지만 낚시를 하는 것을 배우는 것이 여러분의 앞길에 풍요로움이 생기는 것이지요.

본론으로 들어가서..모바일 네트워크를 어려워 하시는 분들이 의외로 많습니다. 생각해 보면 아주 간단합니다. 정말입니다. 자바이기 때문이죠. 물론 C 도 간단합니다. ㅡㅡ;

일단 서버가 자바냐 C 냐 머 이런 차이겠죠..Http 커넥션은 제외합니다. 일단 Http 커넥션은 그냥 해당 URL 커넥션 후에 그냥 받아오기 때문에 네트워크라기 보다는 그냥 단순한 다운로드라고 할 수 있기 때문이죠. 그리고 Http 커넥션은 Http 헤더를 분석하기 때문에 일반 네트워크 보다도 한 5 초 정도 더 시간이 걸립니다. 그러므로 가능한 이용하지 않는 것이 좋다고 생각됩니다. 모바일이 자바이니 서버도 자바면 찰떡 궁합..그렇지 않으면..약간 고생 정도 하겠죠..^^;

먼저 자바일 경우.. 그냥 하면 됩니다 아래 예제를 보시면 알듯이 일단 고하면..됩니다.
문제는 C 일 경우인데..이런 경우도 특별히 다르지는 않습니다. 문제는 그냥 계속적으로 모든 정보를 바이트로 주고 받아야 한다는 것밖에 없습니다. 참 귀찮죠..

자바의 경우에는 네트워크는 쓰레드 처리를 합니다. 왜냐하면 네트워크를 하는 동안에 멈춰

있으면 안되기 때문이죠..가끔씩 얼마 안되는 양일 경우에는 그냥 쓰레드 없이도 하는 경우가 있습니다. 그렇기 때문에 참고로 알려드리는 것입니다.

그럼 자바 모델에 대해서 설명 드리겠습니다.

1. 일단 소켓을 엽니다. IP 와 PORT 를 써야 겠지요

```
// 소켓 열기
socket = (StreamConnection)Connector.open("socket://xxx.xxx.xxx.xxx:2000");
```

2. 데이터를 주고 받기에 가장 편한 Stream 인 Data..Stream 를 이용합니다.

```
        // Data ... Stream..열기
        dataInputStream = socket.openDataInputStream();
        dataOutputStream = socket.openDataOutputStream();
```

3. 그 이후로는 필요한 규칙에 따라서 주고 받으면 됩니다.

무지 간단하죠…

여기서 주의 할 점은..SK-VM 에서는 주고 받는 크기가 정해져 있지 않은 것들에 한해서 미리 읽을 사이즈를 받아야 합니다. 예를 들어 내가 서버로부터 1kb 의 이미지를 받는다고 한다면 서버에서 그냥 이미지를 보내면 클라이언트는 얼마나 받아야 하는지 모르기 때문에 문제가 많이 생깁니다. 그렇다고 1byte 씩 읽으면 속도도 무지 느리죠. 그러므로 먼저 해당 이미지의 크기를 넘겨준 후에 해당 이미지 바이트를 해당 크기 만큼 읽어야 합니다. 참고로 가끔씩 어느 정도의 버퍼를 잡아야지 읽는 속도가 빨라지느냐 하는 야그가 있는데..제가 여러가지 버퍼로 테스트 해본 결과 최대 바이트로 잡는 것이 가장 빠릅니다. 무슨 말이냐 하면 100kb 를 받는다고 하면 버퍼를 100kb 바이트를 잡는 것이 가장 빠르다는 것입니다. 당근 이렇게 클때는 메모리가 별루 없기 때문에 그렇게는 못하지만..일단 이 방법이 가장 빠릅니다. 참고하십시오.

그럼 서버가 C 모델일 경우..

1. 소켓은 똑같이 열겠죠..하지만

```
// 소켓 열기
socket = (StreamConnection)Connector.open("socket://xxx.xxx.xxx.xxx:2000");
```

2. 이럴경우에는 Data..Stream 을 사용하지 못하기 때문에..

```
        // In, out ... Stream..열기
        inputStream = socket.openInputStream();
        outputStream = socket.openOutputStream();
```

일반 스트림으로 연결하신 후에 오직 read(), write()로만 사용하시면 됩니다.

어렵지 않습니다.

그럼 예제를 보면서 이해하도록 하겠습니다.

먼저 예제는 서버가 자바라고 생각하겠습니다.
에뮬레이터에서 1 번 버튼을 누르면 해당 숫자 1 을 서버로 보내서 서버에서는 1 번 버튼에 해당하는 문자를 보내줍니다. 그리고 2 번 버튼을 누르면 이미지를 주고, 3 번 버튼을 누르면 여기서 중요합니다. 3 번 버튼은 문자를 읽어 오는데 UTF 를 사용하지 않고 그냥 read, write 를 이용하여 얻어 왔습니다. 그러므로 주고 받을 때 미리 크기를 주고 받습니다. 참고로 서버 소스도 올리도록 하겠습니다. 그리고 포트는 2000 번을 이용했는데 마음대로 하십시오. 예외 처리나 다른 처리에는 신경을 안썼습니다 .그냥 네트워크에만 집중 했습니다. 이점 양해해 주십시오.

client.java

```
import javax.microedition.lcdui.*;
import javax.microedition.io.*;
import java.io.*;
import java.lang.*;
import com.xce.lcdui.*;
import com.skt.m.*;

public class client extends MIDlet
{
	private Display display;

	// Canvas 화면
	TestCanvas cvs;

	// 네트워크...
	StreamConnection socket ;
	DataInputStream dataInputStream;
	DataOutputStream dataOutputStream;

	//
	public client()
	{
		display = Display.getDisplay(this);

		try
		{
		  // 소켓 열기
			socket =
(StreamConnection)Connector.open("socket://xxx.xxx.xxx.xxx:2000");

		  // Data ... Stream..열기
			dataInputStream = socket.openDataInputStream();
			dataOutputStream = socket.openDataOutputStream();
		  }catch(Exception e)
```

```
		{
			System.out.println("IOEXCEPTION : ....." + e.toString());
		}


	}

	public void startApp()
	{
		cvs = new TestCanvas();
		display.setCurrent(cvs);
	}

	public void pauseApp() { }
	public void destroyApp(boolean unconditional) {   }


	// TestCanvas
	class TestCanvas extends Canvas implements Runnable
	{
		String text;
		Image img;
		boolean isConnect;

		TestCanvas()
		{
			Thread thread = new Thread(this);
		}

		// 일단 NetWork 는 Thread 로 해야 합니다. 그러므로 현재는 Thread 로 구현.
		// 하지만 하는 일이 없습니다. ㅡㅡ;
		public void run()
		{
			while(!isConnect)
			{
				Thread.yield();
			}
		}

		public void keyPressed(int keyCode)
		{
			try
			{
				switch(keyCode)
				{
					// 1 번 키를 누를 경우
					case Canvas.KEY_NUM1:

	System.out.println("Canvas.KEY_NUM1");

						dataOutputStream.writeInt(1);

						text = dataInputStream.readUTF();
						break;
					// 2 번 키를 누를 경우
					case Canvas.KEY_NUM2:
```

```
            System.out.println("Canvas.KEY_NUM2");

                                                    dataOutputStream.writeInt(2);

                                                    int fSize = dataInputStream.readInt();
                                                    byte[] buf = new byte[fSize];

                                                    if(fSize != dataInputStream.read(buf))
                                                    {
                                                            System.out.println("파일 용량
이 다릅니다.");

                                                            break;
                                                    }

                                                    dataOutputStream.writeUTF("잘받았습
니다.");
                                                    dataOutputStream.flush();

                                                    img = Image.createImage(buf, 0,
buf.length);
                                                    break;

                                                // 3 번 키를 누를 경우
                                            case Canvas.KEY_NUM3:

            System.out.println("Canvas.KEY_NUM3");

                                                    dataOutputStream.writeInt(3);

                                                    int bSize = dataInputStream.readInt();
                                                    byte[] bbuf = new byte[bSize];


                                                    if(bSize != dataInputStream.read(bbuf))
                                                    {
                                                            System.out.println("파일 용량
이 다릅니다.");

                                                            break;
                                                    }

                                                    text = new String(bbuf);

                                                    String bText = "아싸라비야...전송 OK";
                                                    byte[] tbuf = bText.getBytes();

                                                    dataOutputStream.writeInt(tbuf.length);
                                                    dataOutputStream.write(tbuf, 0,
tbuf.length);
                                                    dataOutputStream.flush();
                                                    break;

                                                // 4 번 키를 누를 경우 종료합니다.
                                            case Canvas.KEY_NUM4:

            System.out.println("Canvas.KEY_NUM4");
```

```
                                                dataOutputStream.writeInt(4);
                                                break;
                                        }
                                }catch(Exception e)
                                {
                                        try
                                        {
                                                dataOutputStream.close();
                                                dataInputStream.close();
                                                socket.close();
                                        }catch(Exception e1){}

                                        System.out.println(e);
                                }

                                repaint(0, 0, this.getWidth(), this.getHeight() + 16);
                                serviceRepaints();
                        }

                        // 받은 내용들을 화면 보여줍니다.
                        public void paint(Graphics g)
                        {
                                g.setColor(0xffffff);
                                g.fillRect(0,0, this.getWidth(), this.getHeight() + 16);

                                if(text != null)
                                {
                                        g.setColor(0x000000);
                                        g.drawString(text, 0, 0, g.TOP|g.LEFT);

                                        text = null;
                                }


                                if(img != null)
                                {
                                        g.drawImage(img, 0, 0, g.LEFT | g.TOP);
                                        img = null;
                                }
                        }
                }
}
```

너무 간단한 소스죠..도움이 전혀 안될 것으로 알고 있지만 솔직히 네트워크 프로그래밍 자체가 그렇게 어려운 문제가 아니기 때문에 이렇습니다. 단지 예외 처리를 잘해주시면 됩니다. 폰이다 보니깐 가끔씩 네트워크가 안되는 지역은 끊기는 경우가 있는데 이런 경우의 예외처리만 잘해주시면 됩니다. 이번에는 특별난게 없어 간지럽군요. 역시 네트워크는 서로 잘맞는 규약이 중요하다고 생각됩니다. 가끔씩 한글이 깨진다고 하시는 분들도 있는데요..제가 해본 결과 한글이 깨지는 경우는 못봤습니다. 그 이야기는 특별히 설정이 요하지 않기 때문이죠. 한글이 깨지다면 무언가 바이트의 문제가 있을 수 있습니다. 참고로 바이트로 받

은 한글이나 영문을 어떻게 String 으로 보여주냐고 물으신다면..
new String(byte[] byte)를 이용하시면 됩니다. 간단하죠..

다음에는 더 이상의 모든 진도는 끝났다고 생각됩니다. 어떤분이 강좌에 참고하시라고 소스를 하나 주셨는데..게임 소스죠..그것을 참고로 하겠습니다. 게임 프로그래밍에서는 그분이 점 하시는 분입니다. 그래서 소스가 좋을 것 같아서 공개합니다.

그럼 즐건 하루 되십시오..핑크레드.

닉네임 : 핑크레드
성격 : 다소 소심, 다소 삐짐, 다소 이중인격, 다소 수다쟁이
이메일 : pinkred@hanafos.com

추신 : 강좌를 다른 사이트에 올리시는 것은 자유입니다..단지 저한테 메일 한 통 이라도 보내주시고 올리시면 감사하겠습니다. 언제나 좋은 하루 되십시오.