

```

import java.io.File;
import java.util.jar.*;
import player.PlayerI;
import java.net.*;
import edu.neu.ccs.demeterf.demfgen.lib.List;

/** Class responsible for loading the Player Jar/Class, checking, and returning
 * a PlayerI instance */
public class PlayerLoader{
    static void print(String s){ System.out.println(s); }
    static void die(String s){
        System.err.println("!! "+s);
        System.exit(1);
    }
    public static void main(String[] args) throws Exception {
        if(args.length != 1)
            die("usage: java PlayerLoader <path-to-player-jar>");
        load(args[0]);
    }
    static String getAttr(String type, JarFile jar){
        String attr = "";
        try{
            attr = jar.getManifest().getMainAttributes().getValue(type);
            if(attr == null) die("Attribute Empty (" + type + ")");
        } catch (Exception e) { die(type + " attribute not found:\n" + e.getMessage()); }
        return attr;
    }

    public static PlayerI load(String path){
        // Locate and load the Player Jar file
        File file = new File(path);
        JarFile jarFile = null;
        String canonical = path;
        try{
            canonical = file.getCanonicalPath();
            jarFile = new JarFile(new File(path).getCanonicalPath());
        } catch (Exception e) { die("Player Jar not found at " + canonical + "\n" + e.getMessage()); }

        // Check the player class attribute
        String pClass = getAttr("Player-Class", jarFile);
        print(" * Player Class is: " + pClass);

        // Check that the jar is executable (Main-Class attribute)
        String mClass = getAttr("Main-Class", jarFile);
        print(" * Main Class is: " + mClass);

        // Check that the jar is executable (Main-Class attribute)
        List<String> clspth = List.create(getAttr("Class-Path", jarFile).split(" "));
        if(!clspth.contains("../lib/demeterf.jar") && clspth.contains("../lib/gen.jar"))
            die("Necessary Libraries not in the Jar Class-Path");
        print(" * Class-Path: OK");

        // Load the player class, and check its interface
        PlayerI p = null;
        try{
            String jarPath = "jar:file://" + canonical + "!";
            Class<?> cls = new URLClassLoader(new URL[]{new File(jarPath).toURL()}).loadClass(pClass);
            p = (PlayerI) cls.newInstance();
        } catch (Exception e) { die("Failed to load class " + pClass + "\n" + e.getMessage()); }

        // Make sure that Player name is a prefix of the Jar name
        String jarName = file.getName();
        String pName = p.getName();
        if(!jarName.startsWith(pName))
            die("Player name does not match the jar file (" + jarName + "\" != \"" + pName + "\"");

        print(" * " + path + " Looks OK!");
        return p;
    }
}

```