RICH INTERNET TECHNOLOGIES: AN IMPLEMENTATION IN JAVA WEB START IN E-LEARNING

Sidita Duli [1], Klotilda Nikaj[2], Entelë Gavoçi[3]

[1]*Department of Mathematics,University Luigj Gurakuqi, Shkodër, Albania*
[2]*Department of Physics,University Luigj Gurakuqi, Shkodër, Albania*
[3] *Department of Radiation Protection and Monitoring Network, Institute of Applied Nuclear Physics, University of Tirana, Albania*
* siditaduli@yahoo.com

ABSTRACT

Interactive web applications have become very popular in many domains. One of them is e-learning. This article describes in details the implementation of a set of virtual nuclear physics topics in the Java Web Start platform. The set of laboratory examples is chosen based in the high school program in Albania.
This article highlights the role of the Java Web Start platform in the e-learning process.

INTRODUCTION

The e-learning methods help to improve understanding complex matters at saving time needed for their understanding, learning and following training. (Bauer & Fedak, 2010).
A possible approach of e-learning is the interactive web page that allows students to do small experiments by using simulators or solving physical or chemistry problems. It is important that the student can use this web page in any computer platform, from anywhere. In this way, the application becomes easy and familiar in use. Rich internet application (RIA) offers this possibility, using software standard that are widely accepted, also allowing the animations being inserted in the web browser.

During the previous work, is highlighted that Java Applet offers a good environment in creating useful tools in the e-learning process, such as integrated RIA (Duli et al. 2015). But, although available and supported in JDK 9, the Applet API and the Java Plug-in are marked as deprecated in preparation for removal in a future release. Alternatives for applets and embedded JavaFX applications include Java Web Start and self-contained applications. Migrating to Java Web Start provides the ability to launch the application independent of a web browser. When the user is offline or unable to access the browser, a desktop shortcut can launch the application, providing the user with the same experience as that of a native application. The concrete work consists in implementing the laboratories in physics topics in Java Web Start environment.

The goal of this paper is the representation of steps in migration from Java Applets to JavaFX deployed in Java Web Start. It also highlights the opportunities that Java Web Start environment offers in building useful simulations in natural sciences laboratories.

METHODS

JavaFX is designed to provide Java developers with a new lightweight, high performance graphics platform. The intention is for new applications to use JavaFX rather than Swing to build the application's graphical user interface (GUI). This does not mean that Swing is obsolete. With JavaFX, the developer will not have to think a lot about UI widgets and interactions.

The main difference for developers switching from Swing to JavaFX will be getting used to how the graphical components are laid out and the new terminology. A user interface is still built using a series of layers that are contained within a scene graph. The scene graph is displayed upon a top-level container called a stage.
Other notable features with JavaFX 2.0 are:

- The new declarative markup language called FXML. It is based on XML and enables developers to define a user interface for a JavaFX application.
- The new media engine for playing web multimedia content.
- The browser plug-in for loading JavaFX applets.
- The web component for embedding web pages within a JavaFX application.
- The doclet for generating JavaFX API documentation using Javadoc

IMPLEMENTATION AND DISCUSSIONS

The best way to migrate the applet is to rewrite it as a standalone Java application, and then deploy it with Java Web Start technology (Dea et al 2014). Rewriting the applet and testing the resulting application ensures that the converted applet works as expected, and the application can take advantage of the Java Web Start features. JavaFX applications can be run in several ways: as a desktop application from a JAR file or self-contained application launcher, as Java application from the command line using the Java launcher, by clicking a link in the browser to download an application, embedded in a web page when opened.

Java Web Start applications are launched via the Java Network Launching Protocol (JNLP). This involves the deployment of an XML file on the web server with a ".jnlp" file extension and content type "application/x-java-jnlp-file". This file can be linked on a web page and looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<jnlp spec="1.0+" codebase="http://example.org/demo/"
href="physicsproject.jnlp">
 <information>
  <title>Simulacione ne Fiziken berthamore</title>
  <vendor>unishk</vendor>
  <homepage>http://fshn.unishk.edu.al/physicsapplets</homepage>
  <offline-allowed/>
 </information>
 <resources>
  <j2se version="1.7+"
href="http://java.sun.com/products/autodl/j2se"/>
  <jar href="elearning-project.jar" main="true"/>
  <jar href="fiz.jar"/>
 </resources>
 <security>
  <all-permissions/>
 </security>
 <update check="always"/>
</jnlp>
```

By default, applications are packaged into the following files:

- A JAR file, which contains the compiled class files and images.
- A JNLP file, which contains a deployment descriptor for both two web modes: Web Start and embedded in browser.
- An HTML file, which contains basic code for running both the Web Start application and the embedded application using the Deployment Toolkit.
- A web-files folder, which contains an offline set of files from the Java Deployment Toolkit to assist with starting up the application.

To convert an applet to a standalone Web Start application you put the applet's content into a frame and call the code of the former applet's `init()` and `start()` from the `main()` method of the main class. The major work needed is to convert the applet class to the main class of the application. The applet `init` and `start` methods are no longer present; instead, initialize the application at the beginning of the application's `main` method. The JNLP file can be linked directly in a web page and the Web Start application will be launched if Java is installed on the client operating system. Additionally, Oracle provides a JavaScript API called deployJava.js, which can be used to detect whether the required version of Java is installed on the client system and redirect the user to Oracle's Java download page if not.

To quickly begin the migration, the developer should add the `main` method to the original applet class, and then start calling the applet initialization code where it normally gets called from the applet's `init` and `start` methods. When there is a `main` method in the applet class, it will begin launching by using the Java Web Start technology, and then slowly remove the dependency on the `Applet` class and convert it completely to the application's main class.

The following items are things to consider when migrating:

- A Java Web Start application does not run within the web browser. If the applet has any dependency on the browser, the communication code will no longer work. The APIs that are affected include:
- JSObject API for Java-to-JavaScript communication does not work for Java Web Start applications.
- Common Document Object Model (DOM) APIs available for Java Plug-in applets are not available to Java Web Start applications.
- Similar to Java Plug-in technology, for faster start-up performance, Java Web Start technology caches the application JARs and resources downloaded by the application.
- Java Web Start technology provides permanent cookie support on Windows using the cookie store in Internet Explorer (IE), and the cookie-handling behavior is determined by the cookie control in IE. On Linux and Solaris, Java Web Start technology provides permanent cookie support using its own cookie store implementation.

While deploying an applet with the JNLP applet-desc element, it is created using the `AppletContainer` provided by Java Web Start technology. When the applet calls `Applet.getAppletContext()`, it returns the `AppletContainerContext` provided by Java Web Start technology. One advantage of the JNLP extensions mechanism over Java Plug-in technology is that the installed extensions are available to all Java Web Start applications running on the system, no matter what version of the JRE the application is running. For Java Plug-in technology, only applets running in the same JRE version can use the installed extensions.

The rise of web utilization on mobile device browsers, typically without support for plug-ins, prompted the browser developers to restrict or remove standards based plug-in support from their products. This restriction or removal of plug-in support was aimed at

unifying the set of features available across desktop and mobile versions. The Oracle JRE can easily support applets on browsers for as long as the given browser provides the required combination of browser standards based on plug-in API.

As part of the study, there are implemented a set of applications related to the physics experiments, based on the study program of the high schools in Albania. One of these applications simulates the law of radioactive decay. It predicts the number of the not decayed nuclei of a given radioactive substance decreases in the course of time.

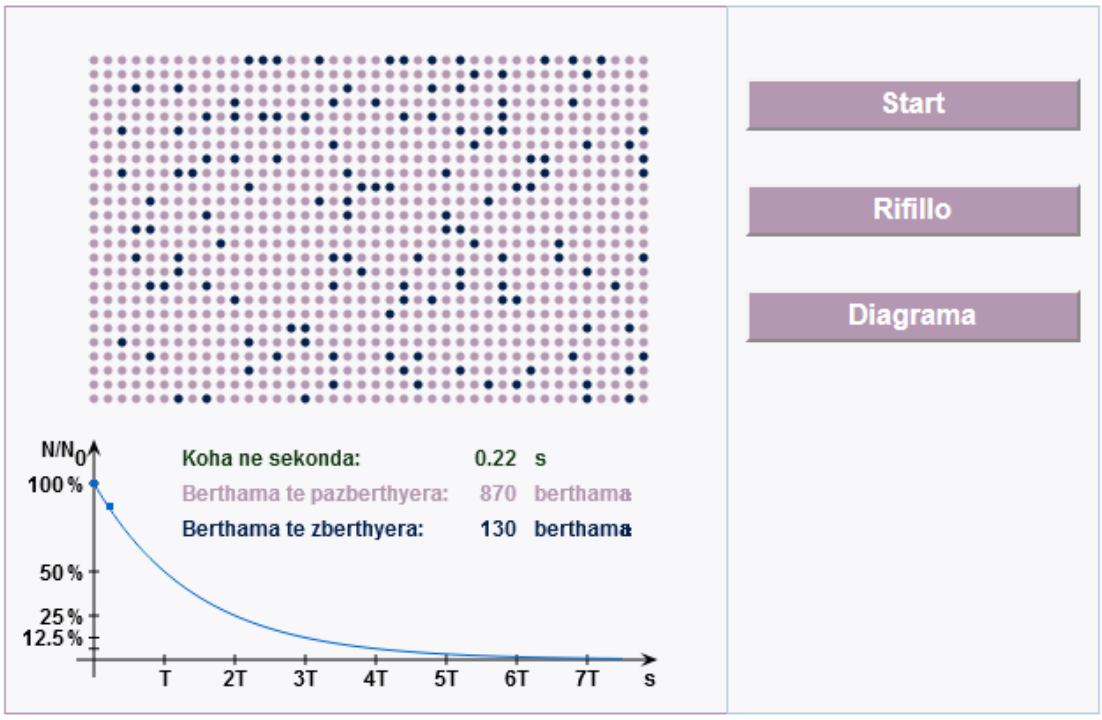

Figure 1: Simulation in Java Web Start of the law of decay

The figure 1 represents a screenshot of the simulation in Java Web Start of the law of decay, during the execution. In the simulation, there are 1000 atomic nuclei of a radioactive substance. The diagram in the lower part of the application represents the fraction of the not yet decayed nuclei (N/N_0) at a given time t. As soon as the application is started with the Start button, the atomic nuclei will begin to "decay". The nuclei which are decayed are represented with black circles. The student interacts by choosing to stop can stop to continue the simulation.

CONCLUSIONS

- Java Web Start offers a good environment in creating useful tools in the e-learning process, such as integrated RIA.
- During the implementation, it is notable the fact of a very clear separation between the model, view, and controller, each of them is a specific file in the deployed form. The view is FXML, the model can be a collection of JavaFX properties, and the controller utilizes dependency injection to have the UI components passed in.

REFERENCES

Bauer, P.&Fedak, V. 2010: Philosophy of Interactive e-Learning for Power Electronics and Electrical Drives: a Way from Ideas to Realization. Journal of Power Electronics, vol. 10, nr. 6: 587-594

Duli,S,.&Nikaj,K. & Gavoçi,E. 2015:Rich internet applications: a comparison between java applets and other similar platforms. 3rd International Conference “Research and Education in Natural Sciences”

Dea,C.&Heckler,M.&Grunwald,G.&Pereda,J.&Phillips,S. 2014 : JavaFX 8: Introduction by Example 2nd Edition, APress