

A Hyperbolic Ontology Visualization Tool for Model Application Programming Interface Documentation

Cody Hyman
Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Dr., Pasadena, CA 91109
August 11, 2011

Abstract:

Spacecraft modeling, a critically important portion in validating planned spacecraft activities, is currently carried out using a time consuming method of mission to mission model implementations and integration. A current project in early development, Integrated Spacecraft Analysis (ISCA), aims to remedy this hindrance by providing reusable architectures and reducing time spent integrating models with planning and sequencing tools. The principle objective of this internship was to develop a user interface for an experimental ontology-based structure visualization of navigation and attitude control system modeling software. To satisfy this, a number of tree and graph visualization tools were researched and a Java based hyperbolic graph viewer was selected for experimental adaptation. Early results show promise in the ability to organize and display large amounts of spacecraft model documentation efficiently and effectively through a web browser. This viewer serves as a conceptual implementation for future development but trials with both ISCA developers and end users should be performed to truly evaluate the effectiveness of continued development of such visualizations.

Contents

Abstract:.....	i
I. Introduction.....	1
II. Background.....	1
ISCA Concept.....	1
Ontologies in ISCA.....	1
III. Objectives	2
An Ontology Viewer.....	2
Issues with Viewing Large Ontologies.....	2
IV. Approach.....	2
Selection of a Viewer	2
Modification of Treebolic.....	3
Main GPL source for Treebolic.....	3
Modification: Modified Linking and Targeting	4
Addition/Modification: Improved Search Functionality.....	4
Modification: Treebolic Generator Upgrades	5
Modification: DTD and Parser Changes	5
Addition: Multi-Language Function Copy to Clipboard	5
Addition: Integrated Tabbed Web browser for simple documentation	5
Addition: XML-RPC SPICE Data Types.....	6
V. Results	7
Two Versions of the ISCA Ontology Viewer	7
Complications	8
VI. Discussion	9
Feedback from NAIF SPICE Developers	9
VII. Conclusion	9
Recommendations for Future Development.....	9
VIII. Acknowledgements	10
IX. References.....	10
Works Cited.....	10

I. Introduction

This paper covers the modification and development of a currently existing hyperbolic tree viewer for the purposes of viewing software application programming interfaces (APIs). The work on this project was conducted over the first 9 weeks of a 10 week summer internship at NASA's Jet Propulsion Laboratory as part of the Integrated Spacecraft Analysis (ISCA) team. As the ISCA project is interested in simplifying the complicated process of simulating spacecraft behavior for planning, sequencing, and analysis, a tool such as the one this paper focuses on may be one of many tools to reduce barriers to understanding models used by future ISCA related systems.

II. Background

ISCA Concept

Currently, all space missions use a large number of activity planning, spacecraft sequencing, and validation simulations as part of what is known as a mission operations system (MOS). The MOS encompasses a wide range of software tools that are more or less tailored to each mission. Currently, there is a large initiative, known as the Advanced Multi-Mission Operations System (AMMOS) that intends to improve MOS reusability while reducing mission costs and individual mission MOS development time. Although an Advanced Multi-Mission Operations System exists on paper and has many components being practically used, it still has incomplete sections that do not meet the AMMOS goals and specifications, particularly in terms of true multi-mission simulation software for mission planning and sequence validation. Additionally, many of the available simulations are used as disparate standalone models with little or no compatibility or interoperability with other models or sequencing tools (1).

Attempts have been made to remedy this problem of a fragmented modeling architecture for years, including the Multimission Spacecraft Analysis Subsystem (MSAS) and a previous attempt at a variant of Integrated Spacecraft Analysis (ISCA). Both of these attempts took on a wide scope project, which involved necessary rewriting of complex spacecraft models, and ultimately ended in failure (2). Recently, a new ISCA 2.0 project was initiated and is still in the early phases of formalizing an operations concept, but aims to avoid the problems plaguing the previous attempts at integrating models into a reusable multi-mission architecture by creating a framework that can integrate existing and future models into the current AMMOS system via an ISCA universal modeling interface (2). While the project has been initiated, it is still years away from having an implemented system.

Ontologies in ISCA

Due to the current early state of ISCA and the models it plans to first integrate it was difficult to find a concrete project. One investigative project, the use of ontologies for code structure visualization, documentation, and code entry was devised, and became

the focus of this paper. Due to the high learning curve and complexity of many of the models used for any number of spacecraft subsystems, it is often taxing for spacecraft planners to adapt these models for use with common planning and sequencing tools used at JPL such as APGEN and SEQGEN. The investigation into using ontologies of these models for ISCA hopes to make it easier for modelers and model adapters to share structural data for the model APIs.

III. Objectives

An Ontology Viewer

The key objective of this project was to create an ontology viewer for looking up organized information on the National Ancillary Information Facility's (NAIF) SPICE API documentation. Currently, this documentation consists of hundreds of web pages, each for a specific function or subroutine, organized alphabetically by name, and includes a set of keywords describing each function. A colleague of mine, Erica Wicks, organized this information into an Extensible Markup Language (XML) based ontology organizing these functions by certain characteristics. A way of visualizing and interacting with this ontology was needed, and thus a project began to implement a custom tree viewer that could readily aid a SPICE user with finding information on each function. With a number of free open-source products readily available, the key objective was to evaluate and select one of these tools and modify it to better suit the application of viewing the SPICE API ontology.

Issues with Viewing Large Ontologies

Given the large number of functions in the SPICE API, viewing an ontological structure containing all of these functions many times over is a difficult process. This is particularly challenging to view as a normal tree structure, as the width of the tree at the leaf (function) level is quite wide. This generally makes it impractical to use a standard tree or even a radial tree in this case.

One common method of displaying large volumes of tree or graph data in a meaningful manner is to use a hyperbolic tree or graph. These structures are functionally similar to any tree or graph structure; however they are mapped on a projection of a hyperbolic space model, such as a Poincare disk (3). An effect of this coordinate system is that the distance between two points grows exponentially as one point moves linearly towards the edge of the display (3). Visually, this allows the user to focus in on the node they intend to look at, along with nearby nodes in detail, while abstracting away structurally distant nodes.

IV. Approach

Selection of a Viewer

Prior to beginning development of a tool, a search was begun for finding off the shelf, preferably cross platform, tools for displaying hyperbolic tree and graph structures. The rough requirements that were sought included a Java based implementation for in-

browser capabilities, a large degree of user interactivity, and the ability to actively display the very large dataset of NAIF SPICE API functions without detriment to performance or user comprehension.

The first program evaluated was HyperGraph, a simple hyperbolic graph viewer that had been previously used by my ISCA colleague, Erica Wicks. After quick evaluation, it was noted that HyperGraph was slow to render the large amount of data included in the SPICE ontology and it was also displayed in a manner that was mostly undecipherable by the user. A comparison of the SPICE ontology with both the HyperGraph and Treebolic versions is shown below in *Figure 1*.

Following the write off of HyperGraph, another hyperbolic tree/graph viewing utility was discovered, Treebolic 2. Written by Bernard Bou, this GNU General Public License (GNU GPL) program, quickly demonstrated that it was easy to use, could render and move nodes in real time without significant latency, and was visually much more understandable for the user. Upon satisfaction of the initial requirements, Treebolic 2 was selected for adaptation to the SPICE documentation viewer prototype project. The modification process following this is described in the next subsections.

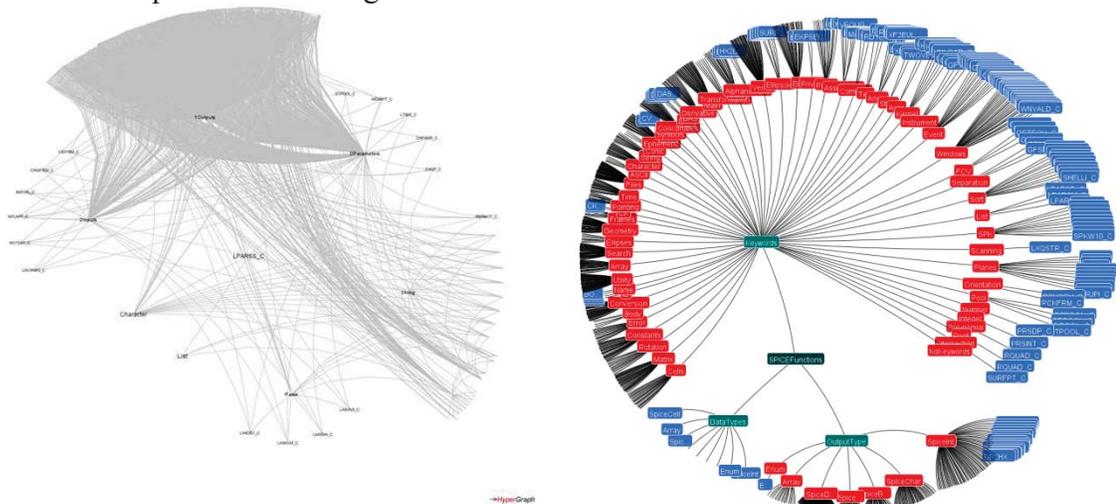


Figure 1: Comparison of Hypergraph (left) with Treebolic (right) displaying the SPICE API

Modification of Treebolic

After selecting Treebolic as a platform to develop on, a large portion of time during the project was dedicated to writing new components for and rewriting many existing components from Treebolic. The entirety of the program source code was written in Java, and released by the developers, and likewise the new classes written for Treebolic were written in Java as well. The remainder of this section discusses the major changes made to transform the Treebolic 2 components into the ISCA Ontology Viewer.

Main GPL source for Treebolic

The GNU General Public License (GNU GPL) that is currently licensing Treebolic allows for modification and redistribution of the source code and compiled programs given they are too under the same license (4). Due to this, it is legally possible to change, add to, and rerelease any of the Treebolic programs by the licensing

terms. The resulting ISCA Ontology Viewer was made by making additions and modifications to the freely distributed Treebolic source code. This licensing also dictates that the ISCA Ontology Viewer is also licensed under the GNU GPL.

Modification: Modified Linking and Targeting

One issue in the original Treebolic program was the lack of hyperlink target functionality. A target is simply an attribute of a link tag that indicates where in a browser the link should be opened. Initially as Treebolic was being developed as an applet, the plan was to open up SPICE documentation pages in an HTML frame; however the applet could only navigate away from the current page rather than loading in the intended frame. The underlying code for loading into a frame was included in the original Treebolic source; however it was not implemented by default. With a few minor changes and additions, the program was enabled to link to the target specified in a node link under the *target* attribute. Later additions to this feature also included a tree level *target* attribute that links to a specified target for the entire tree unless a node contains a link target, in which case the base target will be overridden.

Addition/Modification: Improved Search Functionality

Searching for nodes in a tree was present in the original Treebolic source, however the system and the user interface for the process is non-intuitive and difficult to use. Under this default functionality, the data files being loaded into Treebolic have to specify the ability to use the search feature as well as the search scope to use. To activate the default search, if enabled, one has to open an unlabeled edit box and then specifically search sub-trees in the visualization via a right-click context menu command. Upon finding the results, the viewer simply jumps to the first result node or stayed stationary. No display of the search results or intuitive way of handling multiple positive results or no results existed. To remedy this, I created a new search class along with complimentary GUI components for the new search features.

The GUI components consisted of two parts that were added to the status bar component of the original Treebolic GUI, a search entry section and a search results section. The entry section consisted of a simple text box and combination box for entering a simple search expression and selecting a search limiting scope for *<id>* elements, *<label>* elements, or *<content>* elements. Search results were displayed in a second component which lists all the results from the search and provides a means to jump to a selected node in the viewing pane. By being able to select any of the result nodes, more than one matching node can be navigated to from the search bar.

Behind the new Treebolic search UI components, an underlying search class was constructed as part of the new package *isca.ontology.search*. This class simply consisted of an exhaustive linear search through all of the instantiated node objects in the main tree, checking the specified parameter for a substring containing the search expression. Upon obtaining a result, it was added to a list of search results that are displayed on the results panel at the end of the search. Currently the search class does not have any additional features for sorting or processing the results data. Initial tests with the entire SPICE Ontology, with many hundreds of nodes, the entire search executes in a fraction of a second on a notebook computer.



Figure 2: The ISCA Treebolic Viewer Search Bar

Modification: Treebolic Generator Upgrades

The Treebolic source also contains a GUI based utility program for building and modifying the Treebolic formatted XML data files. This program source was modified and recompiled to include access to new features such as node link targets, tree base targets, and the enabling of the language bar. Not every added feature has been made mutable from the generator program, however this functionality can generally be added in quickly. An example of a feature that has not been made optional yet is the tabbed web browser that operates inside of the main Treebolic viewing widget.

Modification: DTD and Parser Changes

To facilitate new data in the tree model including link targets both at the root level and individual link level, optional feature inclusion, and the new SPICE data type element, the Treebolic Data Type Document (DTD) was modified. This DTD acts as a specification for the types of data that can be parsed by the program and acts as a guide for people or programs creating the XML source files. The Treebolic XML DOM parser class was subsequently modified to parse these new DTD elements and attributes from the XML source during tree initialization.

Addition: Multi-Language Function Copy to Clipboard

As one of the main goals of using an ontology with ISCA was to facilitate the lookup and entry of code from the SPICE API, some facility for doing so was an important feature to implement into the Ontology Viewer. This feature took the form of a multi-language copy method designed to make the search for an implementation of a certain function easier than copying its syntax from a documentation webpage.

The copy method is implemented to work only when preformatted text for the function call is provided in the XML data file for each node. Stored in the `<content>` element of each function node of the demo SPICE XML file is the procedure for a subroutine call in each of the four languages SPICE is released in. In lieu of opening up the documentation page, selecting, copying, and pasting each function header into a separate code editor, the ISCA Ontology Viewer is capable of copying the properly formatted function call with generic parameters by opening the node context menu and selecting the “Copy Function” item. When activated, this triggers a method call to parse the node content for the correct subroutine call in the language selected with a simple radio button based language selection bar. The content parser was implemented in an entirely new Java class contained within the added `isca.ontology.search` source package.

Addition: Integrated Tabbed Web browser for simple documentation

One of the key components to the ISCA SPICE Ontology Viewer is a web browser capable of showing the NAIF SPICE documentation webpages while still maintaining the tree view in the same window. Initially on this project, an applet in a framed web page was used, with one frame containing the applet and another being used for viewing documentation pages.

Using a standard Java Swing component, the JEditorPane, along with an HTMLToolkit, a rudimentary HTML webpage viewer was created. Additional features have been incorporated as well including tabbed browsing, an address bar, a forwards and backwards traversable page history, and a home button currently set to return to the NAIF SPICE main documentation page. As this browser is intended to be used for displaying documentation pages only, the address bar cannot accept new addresses, however following hyperlinks between pages is allowed. This feature was implemented as a new Java package, *html.browser* in the modified Treebolic Source. It is further instantiated as part of the main Treebolic viewing *Widget* class.

The resulting browser is functional, and has no issues in displaying the SPICE documentation pages. Due to some security drawbacks of Java, this feature cannot be used in a web applet version of the viewer without further development.

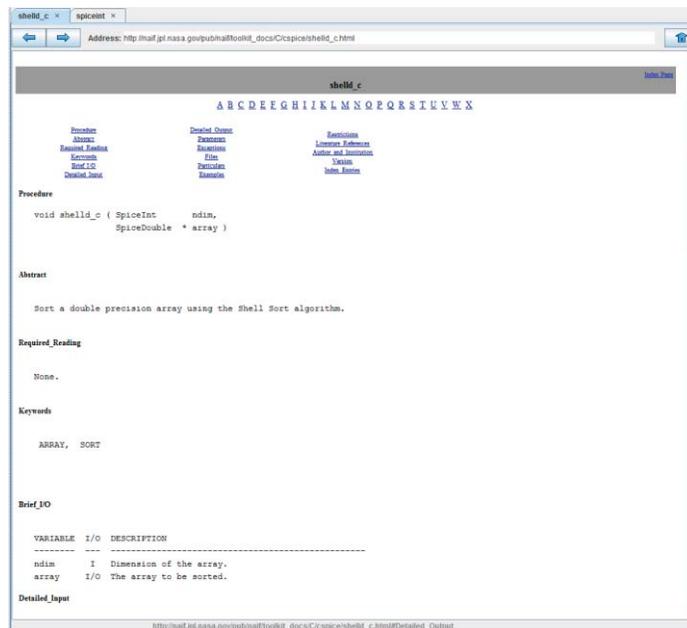


Figure 3: Viewing SPICE Documentation in the Tabbed Browser Component of the Treebolic Application

Addition: XML-RPC SPICE Data Types

Each of the SPICE functions incorporates some number of enumerated data types in the SPICE libraries. With the addition of the `<datatype>` element in the Treebolic Data Type Document, every SPICE function node received specific `<datatype>` elements with links to special documentation pages for each data type used by the function parameters or return values. The pages include basic information on the type and instructions on how to handle these data types in a call to a SPICE server program also being investigated by the ISCA team. In the viewer, the data types show up as special links in the context menu for a function node.

V. Results

Two Versions of the ISCA Ontology Viewer

The resulting creations from this project were two versions of the Treebolic viewer, an in browser web applet, and a standalone Java archive that can be ran as a desktop application. Basic Treebolic functionality works for both platforms, however due to a number of issues discussed in the next subsection, the applet version suffers decreased functionality. Visually, the entire SPICE Ontology could be shown and easily navigated on one screen, however some nodes were hard to see without zooming in on the particularly leaf heavy sub-trees. Additionally, the Treebolic Generator application was updated to add in access to some, but not all, of the other new added features. Images of the resulting viewer programs are both shown below.

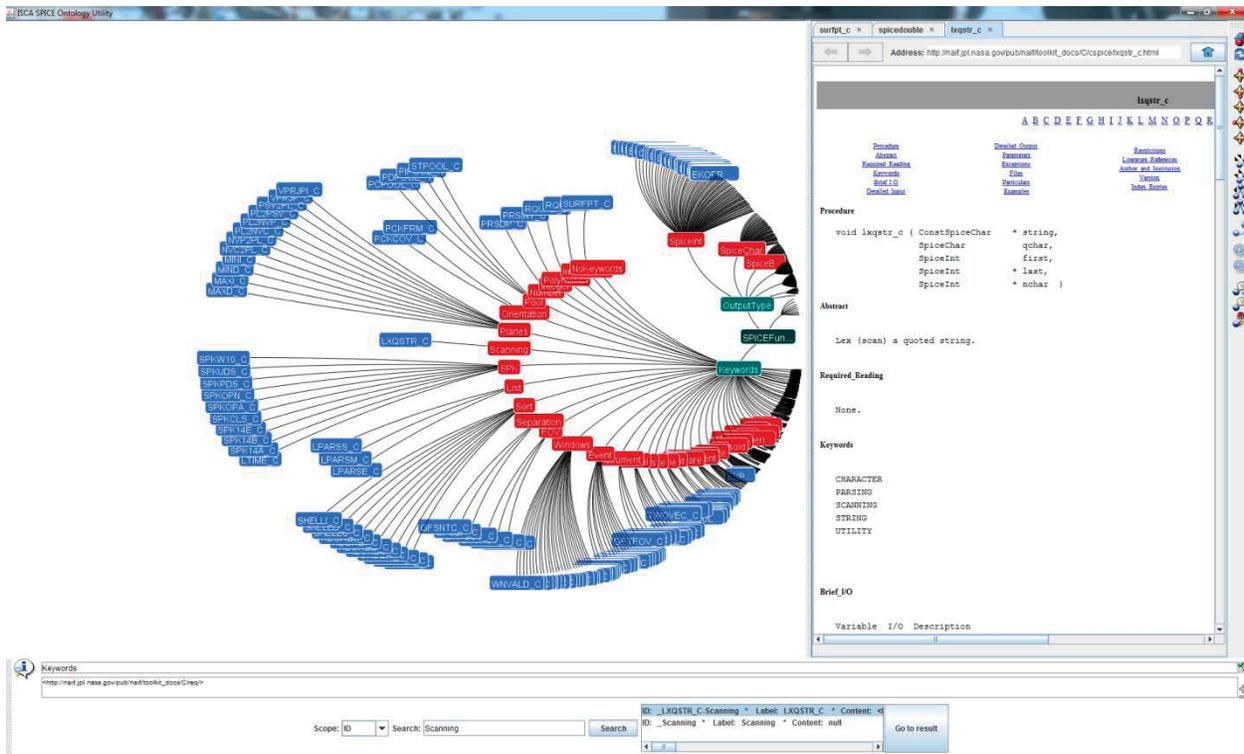


Figure 4: The ISCA Ontology Viewer as a Standalone Application

VI. Discussion

Generally, most of the features that made it into the current version of the ISCA Ontology Viewer function as intended in the unhindered desktop application distribution. While visualization of the SPICE API was made much simpler, this new take on the API still requires some manual searching to find the desired node. Despite the addition of these features, this program still does not strongly link itself to any software development tools and does not yet fit in fully with the intended ISCA use of ontologies for linking the software models with the adaptations in any general structured way.

Feedback from NAIF SPICE Developers

During a demonstration of the ISCA Ontology Utility, members of the NAIF SPICE team expressed interest in possibly enhancing the tool for documentation and learning purposes beyond the organization of functions in the API. While there may have been a limited scope for the initial development of the ontology tool, the SPICE team has suggested that the tool could be made far more useful if it could incorporate extra SPICE information including ephemerides, commonly encountered problems, and various other important SPICE aspects that are not incorporated in the standard API documentation that was used.

As the information and code behind the SPICE library frequently changes, the ISCA Ontology Viewer is in extreme need for more automation on the content creation front. As the XML data source was assembled with information mostly constructed manually, the process may be too slow to maintain in addition to other documentation procedures on a tight schedule. Thus, if more automation in the content creation portion of setting up the viewer could be developed, this tool may have a significantly better chance of being adopted outside of ISCA.

VII. Conclusion

Given the early stages of ISCA and the lack of actual use case test data, it is difficult to determine whether or not continuing the development or use of the Treebolic based ISCA Ontology Viewer will be beneficial to the overall ISCA project. As a program for rendering large amounts of ontological data, the Treebolic component performs excellently and provides extreme portability to this project and possibly many other projects that require the visualization of ontologies.

Recommendations for Future Development

While a functioning version of the ISCA SPICE Ontology now exists, there are still a number of potential development routes worth investigation to improve the usefulness of this tool if it is adopted by the ISCA project or any other group for its intended purpose. The following ideas are reasonable concepts that may be useful paths for future development.

The copying of a procedure call to the system clipboard is just the start of how this tool could be integrated with development platforms for adapting models to other planning, sequencing, and analysis tools. Given the program's Java based code, it could very well be integrated with a cross platform open source integrated development

environment (IDE), such as Eclipse. Likewise, Treebolic could be integrated into other Java based applications beyond development tools.

As ISCA will eventually encompass a large semi-generic framework for spacecraft models other than NAIF SPICE, other ontologies would need to be created for each new model for this Ontology viewer to be useful. As this can often be time consuming, even with parts of the process being automated, more dedicated tools for generating similar ontologies from other models, possibly even from their API source code should be considered.

If the web browser applet version is pursued, it would be applet to enable advanced in-browser functionality. A working alternative to the in-browser applet or the downloadable application may be to use the Java Network Launching Protocol (JNLP) to load the standalone application version from a website.

VIII. Acknowledgements

I would like to acknowledge the following people for their contributions to this project:

Yu-Wen Tung (Mentor) – ISCA
 Pierre Maldague (Co-Mentor) – ISCA
 Erica Wicks – ISCA
 Raymond Wong – ISCA
 Steve Wissler - EPOXI
 Chuck Acton - NAIF
 Nathaniel Bachman - NAIF
 Samantha Krening - NAIF
 Boris Semenov - NAIF
 Edward Wright – NAIF
 Bernard Bou – Creator of Treebolic

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by the Space Grant Program and the National Aeronautics and Space Administration.

IX. References

Works Cited

1. **Chung, Seung.** *Sequence Revitalization Concept of Operations*. [Document] Pasadena : Jet Propulsion Laboratory, 2011.
2. **Maldague, Pierre F and Tung, Yu-Wen.** *Integrated Spacecraft Analysis Operations Concept*. [Document] Pasadena : Jet Propulsion Laboratory, 2011.
3. **Lamping, John, Rao, Ramana and Pirolli, Peter.** *A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies*. [Document] Palo Alto : Xerox, 1995.
4. **Free Software Foundation.** GNU Operating System. *GNU General Public License*. [Online] June 29, 2007. [Cited: August 11, 2011.] <http://www.gnu.org/licenses/gpl.html>.
5. **Oracle.** What Applets Can and Cannot Do. *The Java Tutorials*. [Online] Oracle. [Cited: August 10, 2011.] <http://download.oracle.com/javase/tutorial/deployment/applet/security.html>.