Demo: XML3D – Interactive 3D Graphics for the Web

Kristian Sons
DFKI
Saarbrücken, Germany

Philipp Slusallek
DFKI
Saarbrücken, Germany

ABSTRACT

XML3D is a extension to HTML5 to support interactive 3D graphics in the Browser. XML3D is designed to integrate and interoperate well with ubiquitous W3C standards such as DOM, CSS, and others. Embedding XML3D into this family of standards brings many benefits: It allows millions of exiting web programmers to directly apply their existing knowledge also to interactive 3D graphics. The XML3D proposal and its implementations serve as a demonstrator platform for the W3C Community Group "Declarative 3D for the Web Architecture". In this demo description, we present a selection of features of XML3D, that demonstrate the advantages of a web-based declarative approach to 3d graphics.

Categories and Subject Descriptors

I.3.6 [**Methodology and Techniques**]: Standards—*Languages*; I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Virtual Reality*

Keywords

HTML5, CSS3, XML3D, WebGL, DOM, Web Integration

1. INTRODUCTION

Within the proposed demo we would like to demonstrate XML3D [3] and the XML3D implementations. In all demos, the 3D scene description is embedded inside an XHTML web page and thus available in the DOM. This means, the scene is fully modifiable through the DOM. All interaction with the scene are performed using DOM manipulation, DOM Events and JavaScript.

An important design question is why we decided to design a new description rather than using an existing format. Only few platform and rendering independent 3D description exist. COLLADA is probably the most used and best supported open 3D format, but is designed for exchange of 3D assets. Therefore there are plenty quite complex data structures within the COLLADA specification, with the aim to reduce information loss during exchange. Also the format is not platform independent, but multi-platform capable. A (mostly) platform independent open 3D format is X3D, the successor of VRML. Though there is an integration model into HTML via plug-ins, the development of VRML/X3D and Web Technologies such as HTML, DOM and CSS was autonomous. Thus X3D comes with its own run-time and event model and includes concepts unknown in the DOM/HTML world. A possible approach to these problems is to streamline X3D and to extend it with features such as event attributes and CSS. This has been shown by [1].

The pervasiveness of HTML, CSS and DOM technologies is much higher than for any 3D graphics format. That's the main reason why we rated the compatibility to HTML and CSS higher than backwards compatibility to a specific existing format.

Recently two low-level graphics APIs for the Web popped up, namely WebGL and Stage3D, which provide a wide spread of 3D capabilities in the Browser. However, opposed to a declarative approach, it's not possible to develop 3D web application without a basic understanding of the rasterization pipeline. In our demos we want to show the similarity in developing a XML3D enhanced web application compared to standard web development.

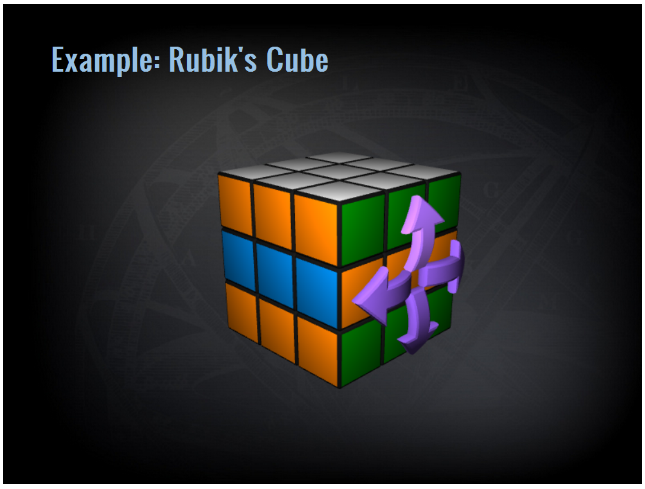


Figure 1: An interactive 3D Rubik's cube embedded in HTML.

2. DEMOS

In the following section we will describe a selection of demos that demonstrate the capabilities of XML3D. The focus for the selection was integration into HTML, interaction methods, debugging of 3D scenes and build Mashups.

2.1 Integration and Interaction

The first demo will show a simple XML3D scene integrated into a HTML5 web page. It shows a Rubik's cube (s. Figure 1). The 3D geometry is described using the XML3D format. It's a simple export from a DCC program. To integrate this XML3D export into a web page, the web author has just to copy the content into an exiting web page. To add interaction to this scene, the author can e.g. add *onmouseover* and *onmouseout* event attributes to the groups or meshes in the scene and execute some arbitrary JavaScript code. The example shows highlighting of the faces changing material attributes. The faces are also configured with *onclick* attributes. If the users clicks on a face of the cube, a direction selection widget is made visible and positioned in the right place. Then again, this widgets has event listeners to finally start rotating one layer of the cube via an animation.

All these actions are simple DOM manipulation tasks and do not require a deep understanding of the 3D graphics pipeline, shaders etc. The DOM manipulations tasks can even be simplified using libraries such as jQuery. This demo uses the JS/WebGL implementation of XML3D and runs in every WebGL-capable Browser.

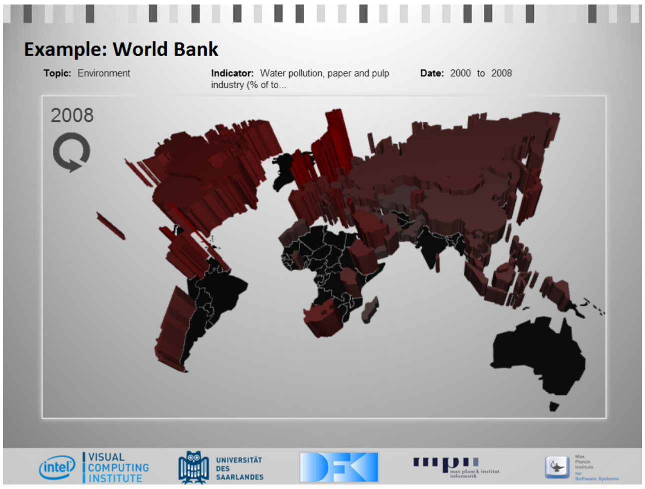
2.2 Rendering independence and Debugging of scenes

The second demo shows three aspects of XML3D: At first, XML3D is not tied to a certain graphics API. For the demo, we use our native XML3D implementation, a modified version of the Chromium Browser that supports OpenGL rendering as well as real-time ray tracing to display the interactive Virtual Museum scene.

Debugging of 3D scenes is still a major issue in all visualization systems. In the native implementations of XML3D, we can use the development tools that come with the Browser to debug the scene. We use the Inspector to select an object in the 3D scene and the inspector highlights the representation in the DOM. It also possible can (de)activate CSS assignments interactively. Web developers can debug their 3D web application using the tools they already know from standard web development. Again, this is possible because XML3D is integrated so tight into HTML and and related technologies.

2.3 Drag and drop and Textures from cross-domains

In the same demo scene, we show that drag and drop is supported in XML3D. The usage is in the same way as web developer know it from standard HTML elements. We show this by dropping an image from a Google Image Search onto a XML3D group with an *ondrop* event attribute. This changes the texture used by the assigned shader.



Example: World Bank

Topic: Environment

Indicator: Water pollution, paper and pulp industry (% of to...

Date: 2000 to 2008

Figure 2: In this demo the World Bank database is queried and results are displayed in the 3D world map.

This action shows another feature of declarative approaches: We have the possibility to use predefined materials. This disables the possibility to read the foreign image from a script using a timing attack. This is an advantage compared to imperative approaches such as WebGL, disallowing usage of cross-domain elements as textures respectively requires CORS approval.

2.4 Mashups and Semantics

Figure 2 shows the third demo, where HTML, XML3D, and XMLHttpRequest is used in combination with a RESTful web service to combine data from an external sources. Each country in the 3D world map is named by it's ISO country code. The user can get the full English country name by hovering over the country, again using simple *onmouseover* and *onmouseout* events in combination with jQuery UI popups.

The user can select a topic, an indicator and a time range from three selection menus. These menus are populated dynamically by querying the World Bank database for available data sets. Once the user has selected a data set, the World Bank is queried for the data sets for all available countries. The result is displayed by adapting the height of the countries and their color.

Information visualization is a good use-case for declarative 3D. Often, no real, but pseudo-3D is required for a good representation of the data. But the development of such visualizations is often easier using real 3D data, because then the projection to 2D is automatically correct.

In this simple demo, the XML3D elements in the DOM are hard-wired with the database entries from the World Bank using the ISO country codes. But one could easily image using RDFa to annotate the geometry representation of the countries. Combined with an ontology this would allow linking data to the 3D geometry in a more flexible and intelligent way. Combining semantic Web technologies and semantic services with XML3D has already been done in a much more complex scenario in ISReal [2].

This demo uses the JS/WebGL implementation of XML3D and runs in every WebGL-capable Browser.

3. REFERENCES

[1] Johannes Behr, Peter Eschler, Yvonne Jung, and Michael Zöllner. X3dom: a dom-based html5/x3d integration model. In *Proceedings of the 14th International Conference on 3D Web Technology*, Web3D '09, pages 127–135, New York, NY, USA, 2009. ACM.

[2] Patrick Kapahnke, Pascal Liedtke, Stefan Nesbigall, Stefan Warwas, and Matthias Klusch. ISReal: An open platform for semantic-based 3D simulations in the 3D Internet. In *Proceedings of the 9th international semantic web conference on The semantic web - Volume Part II*, ISWC'10, pages 161–176, Berlin, Heidelberg, 2010. Springer-Verlag.

[3] Kristian Sons, Felix Klein, Dmitri Rubinstein, Sergiy Byelozyorov, and Philipp Slusallek. Xml3d: interactive 3d graphics for the web. In *Proceedings of the 15th International Conference on Web 3D Technology*, Web3D '10, pages 175–184, New York, NY, USA, 2010. ACM.