

## Wonderblog

Project Wonderland Virtual World Toolkit Blog

### [A Virtual Team Room in Wonderland](#)

Mon, Oct 20, 2008 4:42 PM

For the past eight weeks, Sun Labs in Burlington, MA has had the honor of hosting four teams of interns from Worcester Polytechnic Institute (WPI) in Worcester, MA. The Wonderland project had one of the teams: Joshua Dick and Gerard Dwan, both seniors at WPI. For those who have been inside our MPK20 demo building, our "team room" is quite sparse: besides some pretty graphics, there is in fact, nothing in it. Josh and Gerard's project was to think about what might go in this team room specifically to support the activities of students in the software engineering class at WPI, and build some components to realize that vision. Curious what they came up with? Well, Josh, Gerard, take it away...

Guest blog contributed by WPI Students Joshua Dick and Gerard Dwan. You may watch a video demo about this blog at <http://www.youtube.com/watch?v=IrahHyFTDWA&fmt=18>.

### A Virtual Team Room

We, Josh and Gerard, are seniors at Worcester Polytechnic Institute (WPI) who have been developing components for Project Wonderland for the past two months, as part of our senior project. This post is adapted from a presentation that we gave at Sun Labs after completing the project.

WPI offers a course called Software Engineering, which allows students to emulate working on a team in the software industry. In the course, the professor defines a project for the class, and students form (competing) teams to work on that project. During team meetings, students brainstorm, assess tasks, evaluate their progress and (of course) code! It is the team meeting aspect of this course that our project aims to improve.

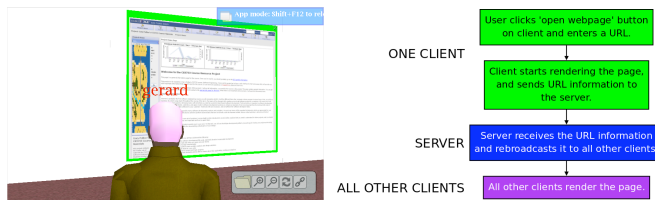
Students don't always have the proper accommodations to conduct meetings. Students come from all over campus, there are (at times) no places to meet, and scheduling is often part of the problem. Using a virtual team room could potentially eliminate all of these issues. Throughout the project, we had a vision of a Virtual Team Room in Wonderland. The Virtual Team Room is a place where students taking WPI's Software Engineering course can meet to collaborate in a 3D space 24x7. They can assess how their team is progressing as well as plan for the future. In addition, course faculty can easily monitor student activity and progress, while offering students relevant advice. Our virtual team room could make use of components that are already built in to Wonderland: things like its phone and voice capabilities, white boards, and PDF Viewers.

So, we started brainstorming ideas for new components that could address our particular set of problems. We thought of things like a virtual library for all of the required reading/on-line materials, task orbs to show project progress, and other (more crazy) ideas.

### HTML Viewer

The first component that we decided to move forward with is the HTML Viewer, which was our 'training wheels' project that we used to get comfortable with developing for Wonderland. The HTML Viewer can display web pages in a way that is far more lightweight than the way it is currently done in Wonderland, with no dependencies on external web browsers/X11. It displays single web pages as an in-world poster, displaying the latest Hudson build statistics or other information in the team room.

The HTML viewer is simple to use, and works similarly to Wonderland's existing PDF Viewer. (Incidentally, the HTML Viewer is based on the PDF Viewer's code.) Users can open web pages, zoom in and out, and refresh pages in their local Wonderland client by using the appropriate HUD buttons. Like the PDF Viewer, the HTML viewer can also be toggled between synchronized and desynchronized modes using the appropriate HUD button. When a page is opened in any client that is in synchronized mode, all other clients that are in synchronized mode begin to render that same page. When in desynchronized mode, a client can open a new web page independently (without affecting other clients,) and can then resynchronize at a later time to see whatever page everyone else is currently seeing. It should be noted that all web page rendering is done on each Wonderland client locally, rather than on the Wonderland server.



We feel that our lightweight HTML viewer is valuable both for our vision of the Virtual Team Room, as well as for Wonderland in general, in situations when a full-featured web browser is unnecessary to simply display a web page passively.

Here are some features we would have liked to add to the HTML viewer if we had more time to work on it:

1. 'Web browser-like' behavior with clickable links
2. Panning/scrolling of web sites, possibly using mouse dragging
3. Refreshing the currently displayed web site on a timer, automatically (The current HTML Viewer does have a manual refresh feature.)

We spent a total of two weeks working on the HTML viewer from start to finish.

### WonderBlocks

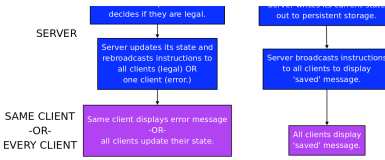
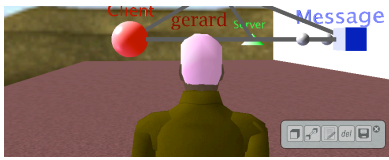
After completing the HTML Viewer, we removed our aforementioned training wheels and did some more brainstorming, considering components that would uniquely utilize Wonderland's 3D space. We came up with an idea that we decided to call WonderBlocks. At first we wanted WonderBlocks to be a utility to display tasks and their dependencies inside the Virtual Team Room. Over time, however, WonderBlocks evolved into a 3D diagramming and data visualization tool for Wonderland. WonderBlocks can be used to display any kind of relational data, from tasks and their dependencies to Flickr tag clouds.

WonderBlocks' current functionality is accessed through its HUD, which is triggered either by clicking any Block or connection, or by simply walking up to the WonderBlocks in-world. There are buttons in the HUD for:

1. Creating Blocks, which can be assigned a name, color, shape, size, and position.
2. Creating connections between Blocks, which can be assigned a name and direction. When in Connect Mode, click two Blocks in succession to connect them.
3. Editing Blocks and connections, which allows the user to change the properties that were assigned to the Blocks/connections when they were first created. When in Edit Mode, click any Block or connection to edit it.
4. Deleting Blocks and connections. When in Delete Mode, click any Block or connection to delete it. Deleting a Block also deletes any connections it's associated with.

The HUD is always manually dismissed by the user by clicking its close button, no matter how the HUD is initially triggered. This way, the user can easily manipulate WonderBlocks from any angle and distance, without the HUD automatically vanishing.





In the future, we hope that Wonderland developers will utilize and continue to improve WonderBlocks. Our ideas for expansion include:

1. Changing the 3D drawing method for scalability. Right now, whenever the diagram changes, the entire diagram is redrawn, rather than only the parts that changed. This may be slow for very large diagrams.
2. 'Prettier' connections utilizing Java3D cylinders rather than unshaded lines.
3. Custom metadata for Blocks / Connections. Right now, the only data that can be associated with WonderBlocks components is the data we outwardly present to the user (color, shape, size, etc.) It would be nice to be able to hover over a Block or connection and see custom, user-assigned data associated with that Block or connection.
4. The ability to drag Blocks in Edit Mode.
5. All user-triggered changes to WonderBlocks diagrams are logged by both the Wonderland client and server, using a consistent format. If some sort of log parser were to be written, Software Engineering course faculty could easily see how students are actually using WonderBlocks, and potentially gauge their work.
6. The ability to create different views of the same data. If WonderBlocks is being used to display project tasks, for example, it would be nice to be able to arrange the diagram by completion status, by deadline, or by owner. This ability largely depends on idea 3 being implemented first.
7. Use WonderBlocks with external data sources.

About idea 7: Currently, WonderBlocks can be used to manually construct 3D diagrams. However, we've taken steps to ensure that WonderBlocks can eventually be used to visualize preexisting data from elsewhere. Picture these potential uses:

1. WonderBlocks could potentially connect to Facebook and log into someone's account, and create a 3D diagram of their Facebook friends and the connections between them.
2. WonderBlocks could connect to the Flickr photography web site, and generate a diagram representing Flickr tags and their relationships (similar tags, etc.) Perhaps clicking on a WonderBlock could display photos associated with the tag (or group of tags) that the Block could potentially represent.
3. For our original vision of the Virtual Team Room for Software Engineering students, WonderBlocks could connect to the SourceForge software development management system and create a diagram of project tasks and their corresponding dependencies and status. Maybe Block shape could represent task type/category, and Block color (green, yellow, and red in this case) could represent completion status of a task.

As you can see, there is a very wide range of possibilities and uses for WonderBlocks.

We spent about five weeks working on WonderBlocks from start to finish.

#### Summary

The HTML Viewer and WonderBlocks are the two components that we created to work towards our vision of a Virtual Team Room in Wonderland. We feel that both components are valuable contributions to the Wonderland community, and we hope that they'll both be used and expanded upon in ways that we never imagined.

The source code for the HTML Viewer and WonderBlocks are available at the following locations in the Wonderland Modules Incubator Subversion repository:

- <https://wonderland-modules-incubator.dev.java.net/svn/wonderland-modules-incubator/trunk/src/modules/apps/2d/htmlviewer>
- <https://wonderland-modules-incubator.dev.java.net/svn/wonderland-modules-incubator/trunk/src/modules/apps/3d/wonderblocks>

AN IMPORTANT NOTE: There are free libraries required for the HTML Viewer that are not included in the repository because of licensing issues. Please see the README file in the HTML Viewer repository for more information.