Building upon the Current Capabilities of WWT

WWT to GitHub

WorldWide Telescope is a complex system that supports a wide variety of research, education and outreach activities. By late 2015, the Windows and HTML5/JavaScript code needed to run WWT will be available in a public (Open Source) GitHub repository. As code moves through the Open Sourcing process during 2015, the **OpenWWT web site** (www.openwwt.org) will offer updated details appropriate for a technical audience, and contact links for additional information.

Leveraging and Extending WorldWide Telescope

The open WorldWide Telescope codebase will provide new ways of leveraging and extending WWT functionality in the future. WWT is already friendly to data and reuse thanks to its extant software development kits, and its ability to import data through both the user interface and "WTML" (WWT's XML based description language to add data into WWT). **The short listing below gives some examples of how data can be accessed, displayed, and explained using WWT as it presently is**. Most of these capabilities are demonstrated quickly in the *"What Can WorldWide Telescope Do for Me?"* **video** at tinyurl.com/wwt-for-me. The www.worldwidetelescope.org/Developers/ site offers resources useful to developers, and details beyond those offered below.

1. Creating Tours

What you can do: You can create a variety of tours with WWT. The tour-authoring interface allows tour creators to guide tour viewers through the Universe by positioning a virtual camera in various slides, and WWT animates the between-slide transitions automatically. Tour creators can also add their own images, data, text, music, voice over and other media to enhance the message. Buttons, images and other elements can link to other Tours, ultimately allowing tour viewers to control their own paths. Tour functionality can be used to create Kiosks, menu-driven multimedia content, presentations, training and quizzing interactives and self-service data exploration. In addition to their educational value, tours can be particularly useful in collaborative research projects, where researchers can narrate and/or annotate various views of data. Tour files are typically small enough to be exchanged easily by email or cloud services. Tours that follow a linear storyline can also be output to high quality video frames for professional quality video production at any resolution desired. Tours can also be hosted in a website to create interactive web content.

Skills Required: WWT tours are one of the most powerful aspects of WWT, and creating them doesn't require any programing skills. You should know what story you want to tell and understand presentation and layout skills. If you can make a PowerPoint presentation then you should be able to make a WWT tour. The WorldWide Telescope Ambassadors (outreach-focused) website provides a good sample of Tours, at wwtambassadors.org/tours, and a good tour to experience to see the largest number of tour features in use all at once is "John Huchra's Universe," at wwtambassadors.org/john-huchras-universe. A sample tour-based kiosk is online at edukiosks.harvard.edu. A video showing a sample research tour (meant for communication with collaborators) is at tinyurl.com/morenessies.

2. Visualize data in WWT Layers

What you can do: You can bring your own data into WWT and position it anywhere in the Universe using the "Layer Manager." Data, including imagery and 2D or 3D catalogs, can be placed on the Earth, in the Sky, or in the 3D Universe. You can use formats such as OGC WKT (Well Known Text) to draw lines, polygons and

create areas or volumes. You can display KML layers, as well as mapping tabular data to different visualizations. Any data with a temporal dimension can be animated over time using a time scrubber. You can import data from flat files, through an Excel plug-in that turns WWT into a graph output type, or through KML, shape files and OData (restful API) sources. You can also define and import orbits around the Sun, Earth, moons, planets or even Lagrange points. If you want to visualize spacecraft, buildings, ships or any other 3D object, you can import 3D models from .obj and .3Ds file formats.

Skills Required: You need to be familiar with your data and understand the basics of working with coordinate systems like RA/DEC, Lat/Long or X,Y,Z. You may need to understand how to massage data in a tool like Excel or through code or scripting languages. You may need skill in doing database queries or using internet sites to extract data files. You should know enough about the data you are visualizing to confirm that unit mappings, and coordinate mappings and scaling are correct. The Excel plugin for WWT can help make the task of importing data into WWT easier (included with the WWT application download).

3. Using the LCAPI (Layer Control API) to feed data into WWT Layers

What it can do: In addition to using the layer manager interactively, you can use scripting and programming languages to send commands and data into WWT from your own code driving the visualization of data in WWT. You can write code that can read data from a database, or create it mathematically and send it to WWT to render. In addition almost all of WWT functionality can be controller through the LCAPI. This means you can create control mechanisms for automated or interactive control of WWT through your own hardware and software. A good example of the use of the LCAPI is the ADS All-Sky Survey as seen in WorldWide Telescope, at adsass.org/wwt/.

Skills Required: You need to be familiar with the programing language or your choice and get the LCAPI documentation and learn how to use HTTP request to send and receive data from WWT. There is also a library called *Narwhal* that provides a .NET wrapper and high level programming features for those who use .NET. Any language that can call HTTP web services can control WWT, and a Python interface has already been implemented, as pyWWT, (see www.jzuhone.com/pywwt).

4. Using MIDI and Xbox gamepad to control WWT

What it can do: MIDI (Musical Instrument Digital Interface) is a protocol for connecting commodity controllers, such as MIDI keyboards, control panels, foot pedals or any other MIDI capable USB device. WWT provides a setup panel for users to map keys, buttons, knobs, sliders and foot pedals to nearly all of WWT functionality. Similarly, WWT allows custom mapping to an Xbox game controller. This functionality can be used in classroom presentations, planetarium control boards, in museum settings, and for data exploration with temporal and other controls.

Skills Required: You need to be able to understand how to connect your MIDI device to your computer, and use the WWT device configuration panel to map your Xbox controller or MIDI device to WWT functions. Note, more custom MIDI devices may require coding or hardware assembly, but off-the-shelf MIDI controllers are plug and play with WWT.

5. Using WWT Tile SDK to process image data

What it can do: WWT offers curated access to many terabytes of image data, but the most important imagery for you might be your own. If you have relatively small amounts of data that fit in single image files, WWT's standalone image-tiler app or on-line image import and tiling service will be all you need, but if

you have very large image sets that cover large swaths of the sky or a planet's surface, or provide terrain or bathymetric data, you may need to construct a software pipeline to process such data to make it efficiently delivered as multi-resolution image and DEM tile pyramids. The Tile SDK is a c# framework with samples that will help you create processing pipelines for large data sets, with TOAST and Mercator outputs as options.

Skills Required: The WWT Tile SDK requires understanding of how to code, compile, debug and run C# applications. You should also have a strong understanding of general imaging including sampling, bitmaps and reading image data from your input sources. You should also understand the projection you intend to have for your input and output sources and know how to properly sample data from them.

6. Modifying and extending the WWT Windows Client

What it can do: The Windows client is the most powerful component, and the most complex. It can power desktop visualizations, power walls, or even full dome multi-projector clusters. For developers with the right set of skills, WWT in Windows can be adapted to an even wider set of requirements.

Skills Required: The WWT Windows Client is built in Visual Studio 2013 with the .net framework 4.5x. It uses custom Winforms controls for the user interface, and DirectX11 for the 3D rendering. The interface between the C# .NET code and DirectX11 is the SharpDX library, another open source component. The client also has significant amounts of HLSL Shader code for both rendering and GP/GPU computation (such as orbits, temporal animation, lighting, etc.). There is a significant amount of TCP/IP networking code, as well as a full HTTP server implementation. To modify the core rendering code you need to know advanced 3D & spherical math and 3D rendering techniques similar to that of 3D Game engines. You also need to understand astronomical mathematics.
There is a layer class model that can be derived from to allow new layer types to be defines without having to modify much of the core WWT codebase. This will be the most popular way of extending WWT, and will probably be exposed as a plug-in model for allowing the main-line installations to take 3rd party WWT layers without having to compile them into the mainline code.

7. The HTML5 Web Client

What it can do: The WWT web client is rich client with a subset of the features of the Windows client, and since it is created as web delivered components it can be extended and reused in a variety of ways. Some of these options don't require modifying the underlying code, just re-using it through WWT's web control API. You can integrate WWT into your data back end to search for and visualize data from databases of imagery, catalogs or simulation data. You can replace or extend the existing UI with your own special purpose UI. You can also add custom rendering of types of data you define.

Skills Required: The User interface for the WWT Web client created in JavaScript with the AngularJS framework. You should be familiar with JavaScript, AngularJS and JSON to modify or extend the web client. The core rendering code is written in C#/.NET and cross compiled to JavaScript using ScriptSharp. There is custom HTML5 Canvas rendering code, with graceful transition to WebGL when available. Working in the core rendering code requires understanding of low-level 3D rendering, 2D and 3D transforms and data structures as well as HTML canvas and WebGL rendering techniques.