Chapter 2: Introduction to Dojo for ArcGIS Server Development

In this chapter we'll cover a number of introductory topics related to Dojo and ArcGIS Server. The ArcGIS Server API for JavaScript is built on top of the Dojo framework so there is a tight integration between the two.

Dojo core is the foundation for all three projects and handles browser normalization, fixes browser incompatibilities, allows DOM querying, remote scripting, drag and drop, data store API, localization and internationalization, Firebug integration, cookie handling, accessibility, and much more.

Dojo takes care of a lot of the lower level functions such as handling core functionality and browser differences. This is important for application developers because it hides a lot of the tedious work involved in testing for browser differences and allows you to focus on developing your GIS applications.

Dijit includes the Dojo framework along with roughly 40 HTML user interface widgets including buttons, text boxes, color pickers, sliders, and many others. Tundra is the default CSS theme for Dijit and is designed to blend into existing color palettes and design. Several additional themes can be used to fit your user interface design needs.

DojoX are Dojo extensions and includes projects such as the grid widget, a graphics library, charting, image handling and more. These often include some very sophisticated projects which are not necessarily as stable as what you'll find in Dojo or Dijit.

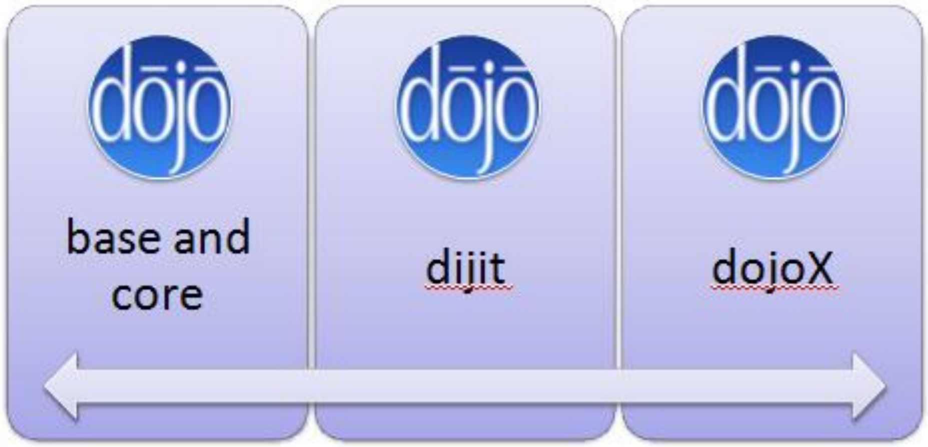

Figure 9: The Dojo Projects

Everything in Dojo is built around Dojo Base which is the foundation for everything else in Dojo. Base provides language and AJAX utilities, a packaging system that allows you to pull in Dojo resources on-the-fly, and many other lower level functions. Core builds on Base by offering additional facilities for parsing widgets, advanced animation effects, drag-and-drop facilities, internationalization, back-button handling, managing cookies, and more. Dijit is built directly on Core and provides an assortment of widgets that you can use to build your user interface. These widgets are very simple to add to a web page. DojoX is a collection of subprojects that are extensions or experimental projects. The grid is perhaps the most commonly used DojoX extension. Finally, Util is a collection of Dojo utilities that includes a JavaScript unit-testing framework and build tools for creating custom versions of Dojo for production settings.

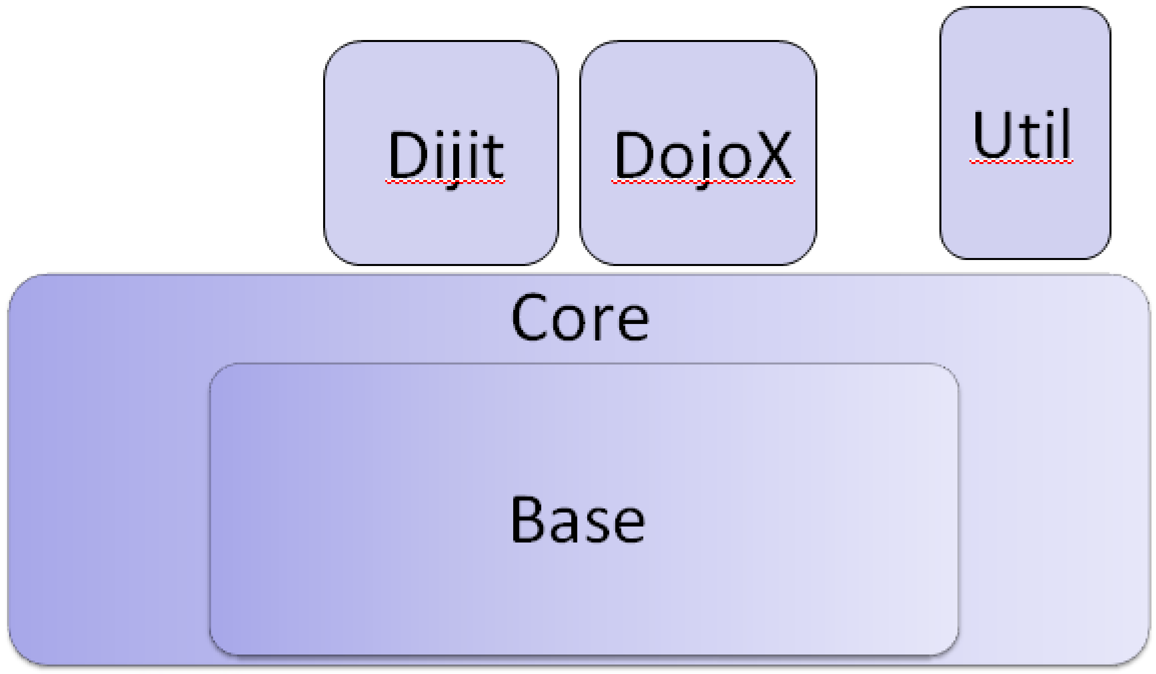

Figure 10: Dojo Architecture

Dijit is one of the primary reasons ESRI chose Dojo. Dijits is a widget system layered on top of Dojo. Using Dijits, you can build amazing Web 2.0 GUI's using very little, or no, JavaScript. You can use Dijit in one of two ways: *declaratively* by using special attributes inside of regular HTML tags, and *programmatically* through JavaScript. Dijits comes bundled with its own themes (tundra, soria, and nihilo) which brings a common design and color scheme to all the widgets. You can override the theme by container or by element to add nuance and flair.

Integrating ArcGIS Server with Dojo
In this section you will learn about the integration of ArcGIS Server with Dojo. First, we'll examine the ArcGIS Server architecture as it relates to Dojo. As you may recall, the ArcGIS Server JavaScript API is built directly on top of Dojo so there is a very tight integration between the two. For each ArcGIS Server JavaScript application that you build, certain boilerplate code must be included. This includes creating references to the API and style sheets as well as the use of dojo.require to import the various resources that you will need to use in your application. Finally, initialization scripts are, as their names implies, JavaScript functions that perform various setup actions for your application on startup.

Dojo
ESRI
ArcGIS JavaScript API

Figure 11: Dojo + ArcGIS Server API for JavaScript

With every ArcGIS Server JavaScript API application that you write a number of lines of boilerplate code MUST be included. There are no exceptions here. Without this boilerplate code nothing else can be accomplished. Essentially there are four steps that you'll need to follow in terms of the boilerplate code that is included. In this chapter I'm going to introduce these steps at a high level. In the next chapter we'll discuss the programmatic implementation of these steps, but for now I just want you to understand them at a high level.

First, in the <head> section of your HTML page you will need to include references to the ArcGIS Server API for JavaScript as well as the Dojo style sheet you intend to use. Remember that referencing the ArcGIS Server API also includes access to Dojo. This reference must be placed in the <head> section of your web page, and also needs to include a version number. This version number refers to the version of the ArcGIS Server API for JavaScript you want to use. Currently this is version 2.6.

Next, in the <body> or a <div> tag in your web page you need to include a reference to the style sheet that was included in the first step. Dojo includes style sheets for the "tundra", "soria", "nihilo", and "claro" themes or you can choose to build your own custom theme. These style sheets are primarily used to control the look and feel of the user input widgets supplied through Dijits.

You then need to add dojo.require statements to import the various ArcGIS Server and Dojo resources that your application will use. Dojo.require acts in a similar way to "include" or "import" statements found in other languages in that it is used to import resources that will be used within the application. Instead of loading all resources at one time, which would hurt the performance of an application, you simply include the resources you need for your application. In our case, we use dojo.require to import both ArcGIS Server and Dojo resources. In the case of Dojo resources you'll often include user interface components found in Dijit or DojoX. Furthermore, you will also need to import the ArcGIS Server resources you need in your application. How do you know which resources to import? We'll examine some of the commonly used ArcGIS Server resources as we move through the chapters. Assuming that you will be creating maps in your application you will always include "esri.map" and from there you'll also need to include other resources as necessary. Dojo.require statements should be placed inside the <head> section of your web page.

After including the boilerplate code that references the ArcGIS Server JavaScript and Dojo API's along with the style sheet information and resources have been imported you will need to create an initialization script that will run immediately after all web page components have loaded. This initialization script is simply a JavaScript function that is used for initial setup operations including variable initialization, creation and centering of your map, loading of datasets, and other common setup tasks. You can give your initialization function any name you'd like, but typical names include "initialize" and "init". The dojo.addOnLoad function is passed the name of your JavaScript function and handles the calling of this function only after all web page components have loaded. It is necessary to wait until all web page components have loaded before running this function because there are times when your initialization function will need access to particular web components and you must ensure that these components have been loaded on the page. Otherwise, errors in your application will occur.

Note: For a more in-depth treatment of the ArcGIS Server API for JavaScript please refer to our Building Custom ArcGIS Server Applications with JavaScript. This course is offered as both a traditional instructor led course and an instructor guided online course. For more information on this course please visit our website at geospatialtraining.com