3 Steps in One Hour

Building a business intelligence application with the ArcWeb Services JavaScript API

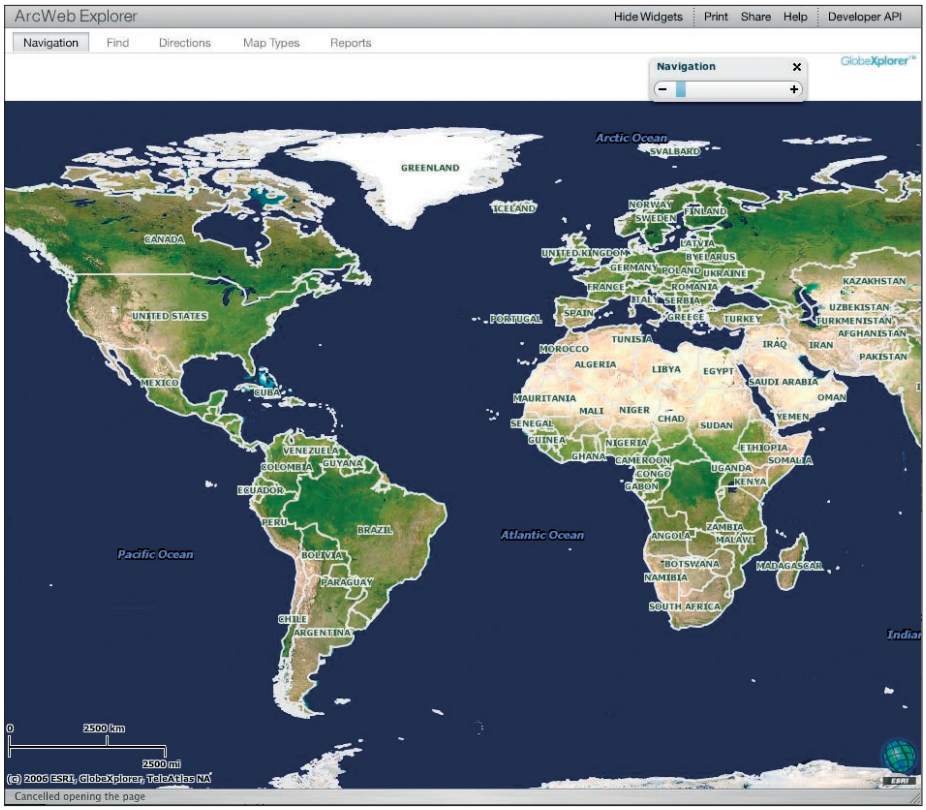
By Andy Gup, ESRI Product Marketing Specialist

Building basic mapping functionality into Web applications is easier today than it has ever been with the availability of free Web services mapping APIs. However, for developers that need to go beyond the basics of placing pushpins and simple overlays into their applications to solve real-world business problems, the options start to get thin. Building complex Web mapping functionality from scratch is often not possible or feasible. The ArcWeb Services JavaScript API provides a platform that comprehensively addresses these issues.

If you are investigating or building a complex Web mapping application, such as one that provides business intelligence, one of the primary challenges is combining information from disparate databases in meaningful and accurate ways. Typically the goal is to build an application that helps end users mine data to discern patterns and meaning that provide a competitive advantage. Common examples of disparate databases include customer lists, point-of-sale information, and physical asset databases such as real estate listings. By combining and analyzing this data using mapping technology, many companies are discovering new business opportunities for existing customers or new product and service offerings targeted at new customers.

The ArcWeb Services JavaScript API is perfectly suited for these situations because its Adobe Flash and Flex-based architecture makes rapid prototyping and deployments very easy. This ability is increasingly important in an environment of tight deadlines and budgets and limited development resources.

To demonstrate how ArcWeb Services can help, this article describes how to build a business intelligence application in three steps that takes less than an hour and uses only HTML and JavaScript. All the listings for this project will be available from *ArcUser Online* (www.esri.com/arcuser). The development steps include building a basic map, adding a searchable data layer, and plugging in functionality via widgets. Once those steps are done, you are ready to launch the application and try it out. If you don't have immediate access to a development environment, you can also try out a free, fully functional version of this application at www.arcwebservices.com/awx.



The basic map

Three Easy Steps

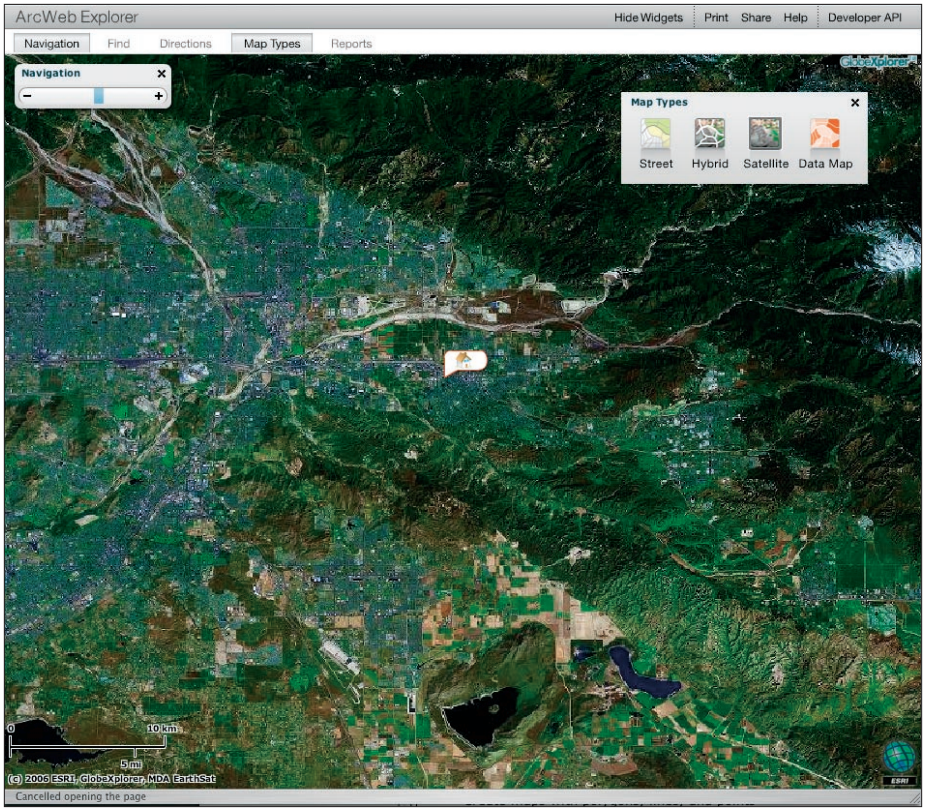
This section provides of quick overview of ArcWeb Services before walking through the steps. The ArcWeb Services JavaScript API is designed for developers who want to prototype and deploy a working Web-based mapping application without spending a significant amount of time developing code or spatially enabling an application or database. The ArcWeb Services JavaScript API provides access to a wide variety of on-demand mapping functionality and content and is a subset of the ArcWeb Services family of APIs that includes SOAP, REST, OpenGIS Location Services (OpenLS), and Java Micro Edition (formerly J2ME). Since all these APIs are included with a standard subscription, you can get immediate access to a wide range of additional functionality using the same license agreement.

The core visualization platform behind the ArcWeb Services JavaScript API is provided by Adobe Flash and Flex. A JavaScript library enables your Web application to control an embedded Flash 8 component. When your application requests a map, ArcWeb Services package it as a SWF file and send it to the client where it runs in the browser's Flash player. The advantage is the ArcWeb Services JavaScript API does all the heavy lifting to integrate your data and the ArcWeb Services mapping data and functionality into a single Rich Internet Application (RIA) framework. The result is desktop-like functionality that provides detailed interaction between your data and the application such as the ability to do spatial searches and comparisons on your data. You will learn more about that and other functionality in the steps that follow.

Basic Preparation

For the sample code to work, you will need

- Listings downloaded from the *ArcUser Online* Web site (www.esri.com/arcuser)



Four different map types provided—Street, Hybrid, Satellite, and Data Map. Click the Map Types tab on the application toolbar to open the Map Types widget. The Hybrid type is shown.

- An ArcWeb Services evaluation account
- An API key
- Registered URL
- Web server
- Integrated development environment (IDE) or Notepad
- Internet access
- Any browser that supports Flash 8 or Flash 9

ArcWeb Services evaluation accounts are good for 90 days and can be obtained at www.arcwebservices.com/evaluate by accessing or creating an ESRI Global Account and completing the ArcWeb Services online registration form. After completing the form, register your URL and obtain an API key at www.arcwebservices.com/v2006/account/account_register_url.do.

You wil also need a basic Web server, such as Apache, and any IDE that can handle HTML and JavaScript. Download a Flash player from www.adobe.com/shockwave/download/alternates/ if you do not already have one. Additional resources are listed at the end of the article.

Step 1—Build the Basic Map

Integrating basic mapping functionality that includes full panning and zooming is the first step. Accomplish this by copying the code from Listing 1, pasting it into an HTML page on the Web server, and pasting the API key you received into the appropriate spot. At this point, you can test launch the application and you should get a basic map.

Continued on page 54

Listing 1: Create a basic map with full pan and zoom capabilities.

```
<html>
<head>
<title>ArcWeb Services JavaScript API Sample App</title>

<script src="http://www.arcwebservices.com/awx/awxapi-1.0.js" type="text/javascript"></script>

<script type="text/javascript">

function onBodyLoad()
{
AWUtils.insertMap("explorer", "<Your API Key Goes Here>");
}
    function onCreationComplete()
    {
       var myExplorer = new AWMap("explorer");
       var myLatLon = new AWLatLon(40.7442, -73.9750);        //set map center
       myExplorer.centerAndScale(myLatLon, 75000);            //set map scale
    }
</script>
</head>
<body onload="onBodyLoad()">
<center>
<div id="explorer" style="width:800px; height:600px;">You need at least <a href="http://www.
adobe.com/shockwave/download/alternates/"> Flash 8 to view this page.</div>
</center>
</body>
</html>
```

3 Steps in One Hour

Continued from page 53

There are five essential items on this page for the application to successfully launch. These are a script element pointing to the ArcWeb Services JavaScript library, an API key, a body element to load the map, the function call to create the map, and a div tag to indicate where to load the map on the page. Try it out. Since this is a basic map with no user interface controls, use your Page Up and Page Down buttons on your keyboard to zoom in and out or click on the map and hold down the mouse button to pan around.

Step 2—Add a Searchable Data Layer

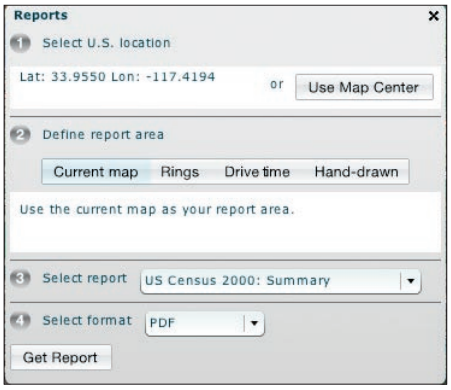
Next add a searchable data layer as shown in Listing 2. This involves assigning a data layer and telling the application how to display the information on the map. For simplicity, the sample application uses the NAVTEQ North American points of interest (POI) data (ArcWeb:NT.POI.NA) and assigns it to a layer that provides query capabilities. "Restaurants" are assigned as the specific POI type. This is one of many datasets that come included with ArcWeb Services.

That's all it takes.

So how does this work? In a traditional Web mapping application, you need a database for storing spatially enabled point data and you have to write the logic for intelligently retrieving relevant points from the database depending on the extent, scale, and other factors related to the map. For instance, if the end user has zoomed out to the city level of New York City, the application logic must retrieve only store locations in New York City.

ArcWeb Services and the AWMap Class addGroupLayer method encapsulate the complexity of building a spatial database and related functionality so that you don't have to build the logic yourself. You don't have to worry about building a spatially aware mechanism that only returns the results that are relevant to the current viewing area. Not having to write retrieval logic significantly reduces the effort it takes to build Web-enabled mapping applications. You can also build more complex queries with custom

Access thematic functionality by choosing Data Map from the Map Types widget.



Use the Business and Demographic Reports widget to generate a report in PDF, XML, or Microsoft Excel format.

datasets such as returning store locations within 20 minutes' drive time from New York City where annual sales are greater than $5,000,000. If you need even greater control, additional functionality is available in other classes, methods, and ArcWeb Services APIs.

To use point data, it must be assigned to a particular data layer. The ArcWeb Services JavaScript API offers six primary group layers including vectorGroupLayer, rasterTileGroupLayer, mapImageGroupLayer, hybridGroupLayer, thematicGroupLayer, and queryGroupLayer. A group layer is a group of several layers that appear and act as a single layer. Group layers make it easier to organize a map and assign advanced drawing order options. This article just focuses on the queryGroupLayer. This layer supports the use of SQL WHERE clauses to filter data and only retrieve and display features that satisfy certain criteria. The count attribute in the queryGroupLayer shown in Listing 2 sets the maximum number of items returned by the WHERE clause to 25 points starting from the closest to the center of the map. At this point in time, the queryGroupLayer only consumes point data and not lines or shapes.

There are four ways to manage point data so that it can be accessed from ArcWeb Services APIs:

- Upload it manually.
- Upload it automatically.
- Choose from one of many datasets included with the API such as the NAVTEQ points of interest data.
- Host the point data on your own server.

The most expedient method for prototyping is to manually upload your data using the Data Manager Web Services available directly through the ArcWeb Developer Portal (Build tab). For applications that require a greater level of customization, you can upload and store lines and polygons, DBF and SHP files, and associated attribute information using the Developer Portal or you can automate data uploads using the ArcWeb Services SOAP API Data Manager Web Service. This data can then be used in map and spatial query data sources.

Step 3—Add Widgets

ArcWeb Services JavaScript API widgets encapsulate rich mapping functionality that can be easily added through JavaScript functions. Widgets included in the API offer functionality, such as create maps, turn layers on and off, find locations, change map styles, convert addresses

```
//Adding a data layer. Use ArcWeb data or to upload your own data go to
//the ArcWeb Portal at www.arcwebservices.com/v2006/builder_main.jsp.
      myExplorer.addGroupLayer("queryGroupLayer", "ArcWeb:NT.POI.NA",
      {whereClause:"FAC_TYPE = '5800'",
      //FAC_TYPE, or Facility Type: Restaurants (5800),
      //Other examples include Hospitals (8060), Police Stations (9221),
      //Hotels (7011), ATMs (3578), Schools (8211)
      labelField:"POI_NAME",           //set the field labels
      orderByField:"POI_NAME",         //set field name display order
      orderByDecending:false,          //
      count:25,                        //set max. number of results returned
      refreshScope:5                   //set buffer around extent
      });
```

Listing 2: Add an interactive data layer.

to coordinates, and create interactive hand-drawn overlays, and have the ability to create business and demographic reports that can be output to PDF, XML, or Microsoft Excel formats.

Again, for brevity, the sample enables all the default widgets, as shown in Listing 3, and makes them available through the application toolbar. A single line of code gives you immediate access to a wide range of functionality.

```
myExplorer.showWidget(AWMap.
WIDGET_WIDGETBAR, 0, 0);
```

Listing 3: Activate all widgets.

The AWMap Class showWidget method invokes a map at a specified x,y pixel location with the origin. This example assigns 0,0 to w,xy, which is the top left-hand corner of the Web page. The WIDGET_WIDGETBAR string is a static constant field representing the WidgetBar widget. If you require greater control over the widgets, there are plenty of examples in the *Online Help Quick Start Guide* and the *Quick Start Plus Guide*.

Widgets are independent Web user interface components that allow you to easily implement behavior that is difficult or impossible to implement in HTML. They save a tremendous amount of development time, especially in the prototyping phase, because you don't have to create functionality from scratch and widgets can be used almost immediately. Widgets can be open and closed or even moved around on top of the application without the need to reload the page. Users can also receive feedback based on their input without needing to wait for a page reload, and multiple widgets can act in concert to display a single page that dynamically responds to user input. You can experiment with this type of functionality by dragging and dropping locations from the Find widget onto the Directions widget.

This is all the coding needed to build the fully functional sample application. If you haven't done so already, download the sample code, install it on your Web server, get an evaluation account and an API key, register your URL, and launch the application. When first launched, the application should display the point of interest data overlaid on top of a basemap.

Functionality Overview

This section provides an overview of the functionality you just built. The application was designed to provide an intuitive visual intelligence tool that accomplishes its tasks by complementing your company's data with a variety of other mapping sources and functionality and that can be tailored to the end user's workflow. The best indicator of this is your feedback on how well ArcWeb Services fits into your environment.

Extensive Data

The data available in the ArcWeb Services APIs is compiled from more than 20 commercial mapping data and content providers. Business intelligence data encompasses thousands of current-year and five-year projections of demographic variables that are updated annually such as 2006 average household income or consumer spending information for various retail goods and services.

Thematics

The business intelligence functionality in this sample is best demonstrated by looking at its thematic mapping and the Report widget capabilities. Thematic mapping displays point data, such as restaurant locations, in relation to areas of demographic information symbolized as color-coded classes and defined by specific areas such as ZIP Codes or census block groups.

To access thematic functionality, select the Map Types tab on the application toolbar to open the Map Types widget. This widget provides a selection of four different map types: Street, Hybrid, Satellite, and Data Map. Choose the Data Map option, and it will open a widget that has more than 120 different demographic categories built in; for example, 2006 Average Household Income. Once you choose this option, thematic data will appear on the map and it will dynamically update as you pan and zoom. The Map Options tab provides controls for altering the geographic boundaries, color ramps, and the number of demographic classes displayed.

Business and Demographic Reports

The Reports widget is used to define areas for creating presentation-ready business or demographic reports using the extent of the current map view, user-defined rings, drive-time polygons, or hand-drawn boundaries. The results can be output to PDF, XML, or Excel. It is easy to create different reports.

Conclusion

If you have followed the steps outlined in this article, you have built a business intelligence application prototype for the Web and now have an idea of the ArcWeb Services capabilities available. The ArcWeb Services JavaScript API is one of five APIs offered for ArcWeb Services. It provides a fast and easy way to build Web-based mapping applications that go beyond basic maps and directions to solve real-world business problems. Because this API is Adobe Flash and Flex based, you can use and reuse its components as building blocks to display large amounts of data and to provide end users with intuitive tools for formatting and even editing the data.

Since not all ArcWeb Services JavaScript API functionality was covered in this article, feel free to check out the resources listed below and also try out some of the other functionality. For example, experiment with the Find widget by uploading the Excel spreadsheet, which is included in the downloadable source code. This simulates yet another way to quickly overlay your point data on top of the application so that it can be used for further analysis. This is especially useful for short-notice presentations requiring access to point data stored on your local machine. The uses are endless, and this business intelligence sample is just the beginning.

The ArcWeb Services JavaScript API is perfectly suited for these situations because its Adobe Flash- and Flex-based architecture makes rapid prototyping and deployments very easy.

Additional Resources

- Download Sample Source Code
- ArcWeb Services Evaluation Account
- ArcWeb Services Explorer
- ArcWeb Services Developer Portal
- ArcWeb Services Online Help
- ArcWeb Services Tutorials and Samples
- ArcWeb Services API Matrix Poster
- *Comparing Vector and Raster Mapping for Internet Applications—an ESRI White Paper*