



Ajax Toolkits

Dojo and jQuery

MIS 4530

Dr. Garrett

Ajax Toolkits

- Toolkits are function libraries written in JavaScript that can easily be used in your scripts
- Each toolkit has its own strengths and weaknesses, so toolkits must be selected carefully
- Each toolkit has its own learning curve and usage requirements
- Popular toolkits are Dojo and jQuery

Dojo

- Dojo is a popular Open Source, portable DHTML JavaScript toolkit that makes it easier for an Ajax developer to build Ajax requests. Why is Dojo called a toolkit? Dojo not only has a rich collection of libraries for JavaScript and Ajax Web development, but also provides an extensive, reusable, and customizable widget system supporting user defined widgets, an enhanced event handling system, and an *io* system which simplifies very complicated data transfers between client and server.

- Dojo is a cross-browser oriented tool which supports various Web browsers, such as IE, Firefox, and Safari. It also solves Ajax browser incompatibility problems in ordinary JavaScript.
- Dojo has a set of powerful JavaScript libraries organized in packages. Dojo 0.4 used to have a layered structure library hierarchy to organize all Dojo functionality in packages. Dojo 0.9 and later versions (currently 1.0) simplify the structure where most Dojo functions (Dojo widgets, events, io and others) are available in Dojo *core* base packages; *dijit* and *dojox* (Extended Dojo project) are sitting on top of Dojo core.

Dojo advantages

- It wraps *XMLHttpRequest* and makes the request construction and configuration much easier.
- It frees developers from detailed configuration of *Ajax XMLHttpRequest*, and from parsing and processing the responses back from the server.
- Its widget system is available in Dojo core and *Dijit* namespace packages.

Ajax XML HttpRequest with Dojo

- A Hello world Dojo Ajax example:

The file “*data.txt*” has a text statement “Welcome to Dojo Ajax!”.

This Web application just downloads the content of this text file, places it in the HTML “div” tag “put_here” placeholder, and displays it.

Assume the dojo core package *dojo.js* library is installed in the *dojo* directory in the root directory under *webapps* in a Web server such as Apache Tomcat.

If you browse this page, the “Welcome to Dojo Ajax” will be displayed in the HTML page.

Here is the HTML file with a *xhrGet* Ajax request.

```
<html>
<head>
  <title>Dojo Ajax</title>
  <script type="text/javascript" src='dojo/dojo/dojo.js'></script>
  <script>
    function welcome() {
      dojo.xhrGet( {
        url: "data.txt",
        // The load function is called on successful response from server
        // It inserts the response to the HTML div "put_here" placeholder
        load: function(response, ioArgs)
          { dojo.byId("put_here").innerHTML = response;
            return response; },
      }
    )
  }
  </script>
</head>
<body>
  <div id="put_here"></div>
</body>
</html>
```

Contd....

```
// The error function displays error message if server does not
// respond right
error: function(response, ioArgs)
    {console.error("HTTP status code: ", ioArgs.xhr.status);
    return response;}
});
}
//Invoke the welcome function when dojo starts up
dojo.addOnLoad(welcome);
</script>
</head>

<body>
<span>
    <div id="put_here" ></div>
</span>
</body>
</html>
```

Three dojo methods are used:

The *dojo.htrGet()* is a request method provided in dojo core package which facilitates *XMLHttpRequest* with GET request method type. *Dojo.htrPost* is another request type method to makes an *XMLHttpRequest*.

List of Dojo supported *XMLHttpRequest* functions, they all take a property object(po) parameter.

- `dojo.xhrGet(po)`
- `dojo.xhrPost(po)`
- `dojo.rawXhrPost(po)`
- `dojo.xhrPut(po)`
- `dojo.rawXhrPut(po)`
- `dojo.xhrDelete(po)`

The *XMLHttpRequest* takes many argument properties. You can pass in a Javascript object, which wraps all necessary properties to the *xhrget* request as shown in the example.

The list of common request property arguments:

- url: *String* type, `"/path/to/myServer.php"`. URL points to server endpoint.
- content: *Object* type, `{name: "value"}`. These properties will be serialized as `name1=value2` and passed in the request.
- timeout: *numeric* type; it wait x milliseconds for the response. If this time passes, then the error callback method is invoked.
- form: `dojo.byId("formId")`; it extracts form content and sends it to server.
- handleAs: *String* type; "text" is default, it can be "json", "javascript", "xml", etc.

- `sync`: *Boolean* type, default is `false`. It indicates whether the request should be synchronous (blocking) or asynchronous.
- `headers`: *Object* type specified in `{}` format. It is used to send additional HTTP headers in the request.
- `load`: `function(response, ioArgs){}`. The load function will be called on a successful response.
- `error`: `function(response, ioArgs){}`. The error function will be called in an error case.
- `handle`: `function(response, ioArgs){}`. The handle function will be called in either the successful or error case.

jQuery

- jQuery takes the approach of providing a large library of functions from a single overriding function
- jQuery also has “plug-ins” that independent developers have created that are freely available
- Like Dojo, jQuery is a large script, making any functions accessible from your JavaScript code

jQuery Examples

- [jQuery accordian](#)
- [Thickbox](#)

Obtaining Dojo & jQuery

- [Dojo](#)
- [jQuery](#)