How to Use Require.js

Include the require.js file into the application main html file. The most important attribute to be included in the script tag is the data-main attribute

**data-main**: It is used by RequireJS to know which module to load in your application.

<script data-main = "libs/main" src = "libs/require.js"></script>

The *scripts/main* is a main JavaScript file of an application that contains the RequireJS configuration.

Configurations of Require.js

* **baseUrl** − It is a route path for all modules which are loaded through RequireJS. The baseUrl is indicated by a string starting with "slash (/)", containing a protocol and ending with ".js" extension. If there is no baseUrl specified, then RequireJS uses the *data-main* attribute path as baseUrl.
* **paths** − It specifies the path mappings for modules which are relative to the baseUrl. It automatically adds the .js extension to a path when mapping the module name.
* **shim** − It provides the usage of non AMD libraries with RequireJS by configuring their dependencies and exporting their global values.
* **map** − For the given module, an application uses same module of different versions for different objectives by sharing their ids to make use of same code for different conditions.
* **config** − It provides the configuration to a module by using the *config*option and this can be done by using the special dependency "module" and calling its ***module.config()*** function.
* **urlArgs** − The query string arguments are used to fetch all resources that are loaded by using RequireJS. It is used for cache busting when there is improper configuration of browser or server.
* **waitSeconds** − It specifies the number of seconds to wait before throwing up on script loading. The default is "7" seconds and "0" disables the timeout.
* **packages** − It provides the CommonJS packages for configuring the loading modules.
* **context** − It provides the name for context loading which allows the loading of different modules in a page.
* **deps** − It is an array of dependencies that is required when Require is specified as config object before loading the RequireJS.
* **callback** − It executes a function after loading the dependencies and is required when Require is specified as config object before loading RequireJS.
* **xhtml** − It is used to create the script elements by using the ***document.createElementNS()*** method when this option is set to *true*.
* **scriptType** − It defines the value for script type attribute used in the document. Default type is "text/javascript".
* **skipDataMain** − It skips the *data-main attribute scanning* while loading the module, if this option is set to *true*.

**Different Ways to Load the Module**

Type1: Defining only Key value Pairs

define({

state: "karnataka",

city: "bangalore"

});

Type2: Defining based on function

define(function () {

//Do setup work here

return {

state: "karnataka",

city: "bangalore"

}

});

Type3: Defining Module as Function with Dependencies

If the module is having dependencies, the placement of the first argument (array of dependency names), the second argument (defining function) and the return object that defines the module is shown in the following syntax

define(["./mnc", "./startup"],

function(mnc, startup) {

return {

state: "karnataka",

city: "bangalore",

addCompany: function() {

mnc.decrement(this);

startup.add(this);

}

}

}

);

Type4- Defining Module as a Function

It is not mandatory for a module to return only objects, any valid value from a function can also be returned. The following syntax is used to define a module as a function –

define(["./mnc", "./startup"],

function(mnc, startup) {

return function(title) {

return title ? (window.title = title) :

startup.storeName + ' ' + mnc.name;

}

}

);

## Type 5- Defining a Module with a Name

In some cases you may have to include a name for the module as the first argument to **define()**. This can be done by using the following syntax

define("js2/title",

["js1/mnc", "js1/startup"],

function(mnc, startup) {

//Define js2/title object in here.

}

);