Store a JavaScript Function on the Server

> **NOTE:**
>
> Do not store application logic in the database. There are performance limitations to running JavaScript inside of MongoDB. Application code also is typically most effective when it shares version control with the application itself.

There is a special system collection named `system.js` that can store JavaScript functions for reuse.

To store a function, you can use the `db.collection.save()`, as in the following example:

```
db.system.js.save(
   {
     _id : "myAddFunction" ,
     value : function (x, y){ return x + y; }
   }
);
```

- The `_id` field holds the name of the function and is unique per database.
- The `value` field holds the function definition

Once you save a function in the `system.js` collection, you can use the function from any JavaScript context (e.g. `eval` command or the `mongo` shell method `db.eval()`, `$where` operator, `mapReduce` or `mongo` shell method `db.collection.mapReduce()`).

Consider the following example from the `mongo` shell that first saves a function named `echoFunction` to the `system.js` collection and calls the function using `db.eval()` method:

```
db.system.js.save(
                  { _id: "echoFunction",
                    value : function(x) { return x; }
                  }
                )

db.eval( "echoFunction( 'test' )" )
```

See http://github.com/mongodb/mongo/tree/master/jstests/core/storefunc.js for a full example.

New in version 2.1: In the `mongo` shell, you can use `db.loadServerScripts()` to load all the scripts saved in the `system.js` collection for the current database. Once loaded, you can invoke the functions directly in the shell, as in the following example:

```
db.loadServerScripts();

echoFunction(3);

myAddFunction(3, 5);
```