# Models - Definition

**API Builder 3.x is deprecated**
Support for API Builder 3.x will cease on 30 April 2020. Use the v3 to v4 upgrade guide to migrate all your applications to API Builder 4.x.

Contact support@axway.com if you require migration assistance.

Place all Model files in the `models` folder. You can only declare one model per file. A Model file is a JavaScript file, which:

1. Loads the `arrow` module.
2. Calls the module's `createModel('name', schema)` method (or another `Model` method), passing in the name of the model as the first parameter and an object defining the model schema as the second parameter.
3. Exports the defined endpoint using the `module.exports` variable.

Set the following keys in the object passed to the `createModel()` method to define the model:

| Name | Required | Description |
|---|---|---|
| `fields` | true | An object that represents the model's schema defined as key-value pairs. The key is the name of the field and the value is the `fields` object. See the next table for details. |
| `connector` | true | Connector to which the model is bound (`string`). Each model can only have **one** connector. Connectors are responsible for reading and writing data from/to their data source. |
| `documented` | false | Determines whether to generate API documentation (`true`) or not (`false`). The default value is `true`. |
| `metadata` | false | Used to provide connector specific configuration (for example, mapping the model to a specific database table for the MySQL connector or defining the join properties). |
| `autogen` | false | Used to determine whether to generate API endpoints directly from the model. The default value is `true`. If the endpoint is auto-generated, you do not need to create an API endpoint definition. |
| `actions` | false | An array of data operations supported by the model. The valid values are: `create`, `read`, `update`, and `delete`. By default, all are supported by the model. |
| `plural` | false | A string used as the property name when your API endpoint returns an array. By default, the plural value is the plural of the model name. For example, if your model is named **car**, the default plural would be **cars**.<br>⚠ This value can be set on an API or a model. |
| `singular` | false | A string used as the property name when your API endpoint returns a single record. By default, the singular value is the name of the model.<br>⚠ This value can be set on an API or a model. |
| `before` | false | One or more blocks to be executed before the request. Blocks are referenced by their `name` property. If you want to execute multiple blocks, you should specify them as an array of block names. If multiple blocks are specified, they are executed in the order specified. |
| `after` | false | One or more blocks to be executed after the request. Blocks are referenced by their `name` property. If you want to execute multiple blocks, you should specify them as an array of block names. If multiple blocks are specified, they are executed in the order specified. |

## Field definition

The `fields` property (mentioned above) supports a number of sub-properties as well. The table below outlines these properties.

| Name | Required | Description |
|---|---|---|
| `type` | true | The field primitive type plus others (for example, string, number, boolean, object, array, date). Type can be any valid JavaScript primitive type. Type can be specified as a string (for example, string) or by the type class (for example, String). |

| | | |
|---|---|---|
| `required` | false | Specifies whether the field is required. The default value is `false`. |
| `validator` | false | A function or regular expression that validates the value of the field. The function is passed the data to validate and should return either `null` or `undefined` if the validation succeeds. Any other return value means the validation failed, and the return value will be used in the exception message. If a regular expression is used, it should evaluate to either true or false. |
| `name` | false | Used if the model field name is different than the field name in the connector's model or the underlying data source for the field name. For example, if my model field is `first_name` and the column in a MySQL database is `fname`, the value of the `name` property should be `fname`. |
| `default` | false | The default value for the field. |
| `description` | false | The description of the field (used for API documentation). |
| `readonly` | false | Either `true` or `false`. If `true`, the field will be read-only and any attempt to write the field value will fail. |
| `maxlength` | false | The max length of the field (specified as an integer) |
| `get` | false | A function used to set the value of a property that will be sent to the client. This property is useful if you want to define a custom field where the value is derived. |
| `set` | false | A function used to set the value of a property that will be sent to the connector. |
| `custom` | `false` | This property should be specified and set to `true` if you are defining a custom field. A custom field is one that does not exist in the underlying data source for the connector you specified. |
| model | false | Model name of the field property. This is either the logical name of a custom model or a connector model name in the form **connector/model_name** (for example, `appc.mysql/employee`) |

## Model schema example

The example below creates the `car` model with the specified schema. The car models will be stored in Mobile Backend Services as CustomObjects. Since the `autogen` property was not set to `false`, API Builder automatically generates the pre-defined endpoints for the client to access the car models using the `<SEVER_ADDRESS>/ api/car` endpoints.

```
var Arrow = require('arrow');

var car = Arrow.createModel('car', {
    fields: {
        make:{type:String, description:'the make of a car '},
        model:{type:String, description:'the model of the car', required:true},
        year: {type:Number, description:'year the car was made', required:true},
        bluebook: {type:Number, description:'kelly bluebook value of the car',
required:true},
        mileage: {type:Number, description:'current mileage of the car',
required:true}
    },
    connector: 'appc.arrowdb'
});

module.exports = car;
```