Please run the application after each step of each practice. The idea is for you to be able to understand step by step every practice.

**Practice 1**

**1. Generate ExtJS Project using Sencha CMD**

D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice1.png

**2. Testing the launch function**

|  |  |
| --- | --- |
| File | Description |
| Application.js | * Implement launch function and put a console.log in it   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice2.png |

**3. Creating a simple view**

|  |  |
| --- | --- |
| File | Description |
| Main.js | * Extend the view from Component * Remove everything and add the initComponent function. * Inside the initComponent function set the html property to “Hello World”   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice1-3.png |

**4. Testing the configs object**

|  |  |
| --- | --- |
| File | Description |
| Main.js | * Add the properties “firstname” and “lastname” to the configs object. * Change the html property to be the result of the concatenation of those properties using the generated getters functions   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice1-4.png |

**5. Testing the apply functions**

|  |  |
| --- | --- |
| File | Description |
| Main.js | * Implement the applyFirstName function.   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice1-6.png |
| Viewport.js | * Remove the items property * Add the initComponent function and create an instance of the Main view using the Ext.create function. * Then, add that instance to the items of the viewport   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice1-6.png   * Now we will create a defer function to test the apply feature. Create an Ext.defer function after creating the Main view instance, in the function, set the firstname of the view   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice1-7.png |

**6. Testing the update functions**

|  |  |
| --- | --- |
| File | Description |
| Main.js | * Create a refresh function that updates the html of the view using the getter functions of the firstname and lastname configs * Implement the updateFirstName and call the refresh function in it.   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice1-8.png |

**7. Testing extjs dynamic loading**

|  |  |
| --- | --- |
| File | Description |
| Main.js | * Create a label in the initComponent function and run the application. You should see a warning in firebug indicating that the label class was loaded on runtime.   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice1-9.png |

**8. Testing statics**

|  |  |
| --- | --- |
| File | Description |
| Main.js | * Add 2 statics to the Main view. For example: SAVE\_URL and REMOVE\_URL   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice1-10.png |
| Application.js | * Create a function called testStatics and print the values of the statics in there.   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice1-11.png |

**9. Testing singletons**

|  |  |
| --- | --- |
| File | Description |
| ConfigManager.js | * Create a class called ConfigManager.js in the app folder * Add a function to that class called loadConfigurationFile that will only have a console.log line. * Set the singleton property to true   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice1-12.png |
| Application.js | * Add the new class to the requires array * Create a function called testSingleton and call the loadConfigurationFile in it.   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice1-13.png |

**Practice 2**

**1. Creating a TabPanel view using xtypes**

|  |  |
| --- | --- |
| File | Description |
| MyFirstView.js | * Create a view called MyFirstView that extends from Ext.tab.Panel and have 2 containers as items   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice2-2.png |
| Application.js | * Add the view to the views collection of the Application file.   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice2.png |
| Main.js | * Add the new view as a child of the Main view   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice2-3.png |

**2. Testing the hbox layout**

|  |  |
| --- | --- |
| File | Description |
| MyFirstView.js -> First tab | * Set the layout to hbox and align to stretch * Add 2 containers as childs. * Set the style property to each container to set different background colors to each one so we can differentiate them * Set the flex property to 1 to each container * Set the height property like this: First container = 200 and Second container = 150 * Play changing the align property to their other values: top, middle and bottom. Also changing the flex values.   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice2-4.png |

**3. Testing the vbox layout**

|  |  |
| --- | --- |
| File | Description |
| MyFirstView.js -> Second tab | * Set the height to 400 * Set the layout to hbox and align to stretch * Add 2 containers as childs. * Set the style property to each container to set different background colors to each one so we can differentiate them * Set the flex property to 1 to each container * Set the width property like this: First container = 200 and Second container = 150 * Play changing the align property to their other values: left, center and right. Also changing the flex values   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice2-5.png |

**4. Testing the fit layout**

|  |  |
| --- | --- |
| File | Description |
| MySecondView.js | * Create a new view called MySecondView that extends from Container with layout fit, height of 400 and a single item with background color. * Remember to add the view to the Application.js views collection   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice2-6.png |
| Main.js | * Add the view as a child of the main container   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice2-7.png |

**5. Testing the card layout**

|  |  |
| --- | --- |
| File | Description |
| MySecondView.js | * Change the layout to card * Add 2 containers as childs with different background color * Add a defer function in the initComponent function to change the active item of the card layout   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice2-8.png |

**6. Testing docked ítems**

|  |  |
| --- | --- |
| File | Description |
| MySecondView.js | * Extend from Ext.panel.Panel * Add 2 toolbars as docked items. One at the top and one at the bottom * Add one container as a child with background color and height of 100%   D:\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice2-9.png |

**Practice 3**

**1. Creating a model**

|  |  |
| --- | --- |
| File | Description |
| Person.js | * Create a class called Person in the model folder. * Add 4 fields to the class: Id int, firstname string, lastname string and age int. * Remember to add the model to the Application.js models collection   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-1.png |
| Application.js | * Create a function called testPersonModel and call it from the launch function. * Create an instance of the Person model with some values and print them in the console.   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-2.png |

**2. Testing date fields**

|  |  |
| --- | --- |
| File | Description |
| Person.js | * Add a date field called birthdate with dateFormat = ‘d/m/Y H:i:s’   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-4.png |
| Application.js -> testModelPerson | * Add a date field to the Person instance   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-3.png |

**3. Testing the convert function**

|  |  |
| --- | --- |
| File | Description |
| Person.js | * Add a field called fullname that has a convert function that returns the concatenation of firstname and lastname   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-5.png |
| Application.js -> testModelPerson | * Print the value of fullname   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-6.png |

**4. Testing hasMany assocations**

|  |  |
| --- | --- |
| File | Description |
| Address.js | * Create a new model called Address. * Add 4 fields: id int, address string, country string and phone int. * Remember to add the new model to the Application.js models collection   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-7.png |
| Person.js | * Add a hasMany association to the Address model   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-8.png |
| Application.js -> testModelPerson | * Add a new address to the Person instance and print the length   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-9.png |

**5. Testing model validations**

|  |  |
| --- | --- |
| File | Description |
| Person.js | * Add the following fields:   + gender -> string   + username -> string   + email -> string   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-10.png   * Add the following validations:   + firstname = presence and length of 2   + gender = inclusion of values male and female   + username = exclusion of values admin and operator   + email = email with custom error message   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-11.png |
| Application.js | * Create a function called testValidations * Create a new instance of a Person with invalid values * Call the validate function * Show the errors in an alert message   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-12.png |

**6. Creating a store**

|  |  |
| --- | --- |
| File | Description |
| Persons.js | * Create a new store called Persons in the store folder * Set the Person model to the store * Set some inline data   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-13.png |
| Application.js | * Create a function called testPersonsStore * Get the store instance using Ext.getStore function * Print each record in console   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-14.png |

**7. Testing store custom functions**

|  |  |
| --- | --- |
| File | Description |
| Persons.js | * Create a function called getWomen in which we will loop through the items and look for the persons with gender = female   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-15.png |
| Application.js -> testPersonsStore | * Print in the console the values returned by the getWomen function   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice3-16.png |

**Practice 4**

**1. Creating a form view**

|  |  |
| --- | --- |
| File | Description |
| MyFirstForm.js | * Create a new view called MyFirstForm in the views folder * Extend from Ext.form.Panel * Set the following properties:   + bodyPadding = 10   + defaultType = textfield * Add 3 items:   + firstname   + lastname   + age * Remember to add the new view to the Application.js views collection   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice4-1.png |
| Main.js | * Change the item of this container to be the form we just created   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice4-2.png |

**2. Testing the anchor layout**

|  |  |
| --- | --- |
| File | Description |
| MyFirstForm.js | * Set the layout to anchor * Set each item anchor to 100%   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice4-3.png |

**3. Testing default values**

|  |  |
| --- | --- |
| File | Description |
| MyFirstForm.js | * Set the defaults object with anchor = 100% * Remove the anchor property of each item   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice4-4.png |

**4. Testing form field validations**

|  |  |
| --- | --- |
| File | Description |
| MyFirstForm.js | * Add the following properties to the firstname textfield:   + minLength = 2   + allowBlank = false   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice4-5.png |

**5. Adding a combobox to the form**

|  |  |
| --- | --- |
| File | Description |
| Genders.js | * Create a new store called Genders in the store folder * Set the fields property to: name and value * Add some inline data * Remember to add the new store to the Application.js stores collection   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice4-7.png |
| MyFirstForm.js | * Add a combobox field to the form. * Assign the genders store to the combobox   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice4-6.png |

**6. Adding a dateField to the form**

|  |  |
| --- | --- |
| File | Description |
| MyFirstForm.js | * Add a dateField to the form with dateFormat = ‘d/m/Y’   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice4-10.png |

**7. Copy and paste Person model from Practice 3**

|  |  |
| --- | --- |
| File | Description |
| Person.js | * Copy and paste the Person model from Practice 3 into the model folder * Leave the following fields and remove the rest: id, firstname, lastname, age, gender, email and birthDate * Change birthDate dateFormat to ‘d/m/Y’ * Also remove associations * For the moment leave the validations as they are * Don’t forget to add the Person model to the Application.js models collection |

**7. Filling the form with a model instance**

|  |  |
| --- | --- |
| File | Description |
| MyFirstForm.js | * Implement the initComponent function * Create a new instance of a Person model with some values * Obtain the formBasic instance with the getForm() function * Call the loadRecord function passing as parameter the Person instance created before   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice4-11.png |

**8. Improving Person model validations**

|  |  |
| --- | --- |
| File | Description |
| Person.js | * Leave the following validations and remove the rest:   + firstname length, min 2   + gender inclusion, values = Male and Female   + email email   + birthDate presence * Add a user friendly message to each validation   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice4-12.png |

**9. Testing form markInvalid**

|  |  |
| --- | --- |
| File | Description |
| MyFirstForm.js -> initComponent | * Create a defer function after the loadRecord function * Fill the model with the form values * Validate the model and mark the form as invalid in case is necessary * Test creating a Person instance with invalid values   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice4-13.png |

**10. Adding validations on the firstname field**

|  |  |
| --- | --- |
| File | Description |
| MyFirstForm.js | * Add the following properties to the firstname textfield:   + minLength = 2   + minLengthText = Firstname is requried and its min length is 2   + allowBlank = false   + allowBlankText = Firstname is requried and its min length is 2   + validateOnChange = true   + validateOnBlur = false * Alternate the values of validateOnChange and validateOnBlur and check the difference   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice4-14.png |

**11. Creating a custom validator**

|  |  |
| --- | --- |
| File | Description |
| Application.js | * Implement the launch function * Add the following vtype to the extjs default vtypes   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice4-15.png   * Add 2 textfields just to test the custom vtype   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice4-16.png |

**Practice 5**

**1. Creating a grid panel view**

|  |  |
| --- | --- |
| File | Description |
| Simpsons.js | * Create a store called Simpsons in the store folder * Add the fields: name, email and phone * Add some inline data * Remember to add the new store to the Application.js stores collection   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-1.png |
| MyFirstGrid.js | * Create a view called MyFirstGrid in the views folder * Extend from Ext.grid.Panel * Add 3 columns: name, email and phone * Assign the Simpons store   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-2.png |
| Main.js | * Add the MyFirstGrid as the only child of the main container view   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-3.png |

**2. Creating a boolean column**

|  |  |
| --- | --- |
| File | Description |
| Simpsons.js | * Add the field isAlcoholic   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-4.png |
| MyFirstGrid.js | * Add a Boolean column linked to the isAlcoholic field * Set the trueText and falseText properties   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-5.png |

**3. Creating a date column**

|  |  |
| --- | --- |
| File | Description |
| Simpsons.js | * Add the birthdate field   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-7.png |
| MyFirstGrid.js | * Create datecolumn with format = ‘Y-m-d’   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-8.png |

**4. Creating a number column**

|  |  |
| --- | --- |
| File | Description |
| Simpsons.js | * Add the weight field   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-9.png |
| MyFirstGrid.js | * Create numbercolumn with format = ‘0.00 Kg’   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-10.png |

**5. Creating template and rownumberer columns**

|  |  |
| --- | --- |
| File | Description |
| MyFirstGrid.js | * Create a template column with the following template: ‘{name} Simpsons’   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-11.png   * Create a rownumberer column   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-12.png |

**6. Creating an action column**

|  |  |
| --- | --- |
| File | Description |
| MyFirstGrid.js | * Create an action column with 2 items: Edit and remove. * Add a handler to each item and alert the record that was clicked   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-13.png |

**7. Creating a column renderer**

|  |  |
| --- | --- |
| File | Description |
| MyFirstGrid.js | * Create a column with a renderer function   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-14.png |

**8. Cell and Row Editing**

|  |  |
| --- | --- |
| File | Description |
| MyFirstGrid.js | * Set the selType to rowModel * Add the RowEditing plugin * Add an editor to the name column   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-15.png   * Change the selType to cellmodel * Change the plugin to CellEditing   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice5-16.png |

**Practice 6**

**1. Copy and paste MyFirstGrid and Simposons store from Practice5 and create a Simpson model**

|  |  |
| --- | --- |
| File | Description |
| Simpson.js | * Create a Simpson Model with the following fields:   + id int   + name string   + email string   + phone string   + isAlcoholic Boolean   + birthDate date, dateFormat = Y-m-d   + weight number * Remember to add the new model to the Application.js models collection   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-1.png |
| Simpsons.js | * Copy and paste the Simpsons store from Practice5 * Remove the fields and data properties * Set the model to “Practice6.model.Simpson” * Set autoLoad to true * Remember to add the new store to the Application.js stores collection.   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-2.png |
| MyFirstGrid.js | * Copy and paste MyFirstGrid view from practice 5 * Leave the following columns and remove the rest: name, email, phone, isAlcoholic, birthdate and weight * Remove plugins and selType * Remove the editor of the name column * Remember to add the view to the Application.js views collection   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-3.png |
| Main.js | * Set the MyFirstGrid as the only child of the main container   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-4.png |

**2. Creating an Ajax proxy to load data from a local json file**

|  |  |
| --- | --- |
| File | Description |
| Simpson.js | * Set an ajax proxy to the Simpson model with url = resources/data/simpsons.json * Review the simpsons.json file * Set a proxy reader of type = json and root = simpsons   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-5.png |

**3. Creating a localStorage and sessionStorage Proxies**

|  |  |
| --- | --- |
| File | Description |
| Animal.js | * Create a model called Animal with 2 fields:   + id int   + name string   + group string * Set a localStorage proxy   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-6.png |
| Application.js | * Implement the launch function * Create an instance of Animal model with some values and save it   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-7.png |
| Animal.js | * Change the proxy to a SessionStorage proxy and test again   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-8.png |

**4. Reading data from a server using a JSONP proxy**

|  |  |
| --- | --- |
| File | Description |
| RottenMovie.js | * Create a model called RottenMovie with the following fields:   + id int   + title string   + year int * Set a json proxy with url = <http://api.rottentomatoes.com/api/public/v1.0/movies.json?q=fight+club&page_limit=10&page=1&apikey=jv7ktqjfbzj9rxe2gkx5fvk5> * Set the json reader with root = movies * Remember to add the new model to the Application.js models collection   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-9.png |
| RottenMovies.js | * Create a store called RottenMovies * Set the model to “Practice6.model.RottenMovie” * Set autoLoad to true * Remember to add the new store to the Application.js stores collection   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-10.png |
| RottenMoviesGrid.js | * Create a view called RottenMoviesGrid * Extend from Ext.grid.Panel * Add the following columns: id, title and year * Set the store to ‘RottenMovies * Remember to add the new view to the Application.js views collection   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-11.png |
| Main.js | * Add the RottenMoviesGrid to the main container   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-12.png |

**5. Adding pagination to the RottenMoviesGrid**

|  |  |
| --- | --- |
| File | Description |
| RottenMovie.js | * Add the following properties to the proxy:   + pageParam = page   + limitParam = page\_limit   + startParam = undefined * Add the following properties to the proxy reader:   + totalProperty = total   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-13.png |
| RottenMovies.js | * Add the following properties:   + pageSize = 10   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-14.png |
| RottenMoviesGrid.js | * Add a pagingtoolbar component to the grid   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice6-15.png |

**Practice 7**

**1. Copy and paste MyFirstForm view, Genders store and Person model from Practice4**

|  |  |
| --- | --- |
| File | Description |
| Genders.js | * Copy and paste Genders store from Practice4 * Remember to add the new store to the Application.js stores collection |
| MyFirstForm.js | * Copy and paste MyFirstForm view from Practice4 * Remove the initComponent function * Remember to add the new view to the Application.js views collection |
| Main.js | * Add the MyFirstForm view as the only child of the main container |
| Person.js | * Copy and paste the Person model from Practice 4 |

**2. Adding save and reset buttons to the form**

|  |  |
| --- | --- |
| File | Description |
| MyFirstForm.js | * Add the buttons save and reset to the form   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice7-2.png |

**3. Creating a controller**

|  |  |
| --- | --- |
| File | Description |
| MyFirstFormController.js | * Implement the init function and add listener to the afterrender of the form   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice7-3.png   * Add the following refs:   + myfirstform = myfirstform   + resetFormButton = myfirstform button[itemId=btnResetForm]   + saveFormButton = myfirstform button[itemId=btnSaveForm]   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice7-4.png   * In the afterrender handler call a function that will fill the form with a Person instance   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice7-5.png   * Add the reset and save listeners:   + In the reset handler call the reset function like this form.getForm().reset()   + In the save handler, validate the model and mark the form as invalid in case is necessary   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice7-6.png |

**4. Creating custom events**

|  |  |
| --- | --- |
| File | Description |
| MyFirstForm.js | * Fire custom events when save and reset buttons were clicked   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice7-7.png |
| MyFirstFormController.js | * Change the way in which save and reset actions are listened   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice7-8.png |

**Practice 8**

**1. Creating a template for a container**

|  |  |
| --- | --- |
| File | Description |
| Main.js | * Set a container with itemId = templateContainer as the only child of the main container   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice8-1.png |
| MainController.js | * Add a property called templateData with some example values   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice8-2.png   * Add a listener to the afterrender of the templateContainer and there, create an XTemplate with the templateData and override the templateContainer content with the result of the template   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice8-3.png |

**2. Template conditional processing**

|  |  |
| --- | --- |
| File | Description |
| MainController.js | * Add a hasVacations property to the templateData and add an “if” condition to the template   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice8-4.png  D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice8-5.png |

**3. Implementing functions in templates**

|  |  |
| --- | --- |
| File | Description |
| MainController.js | * Add the property templateData2 with some values   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice8-6.png   * Create a new template that has a function inside that concatenates fname + lname   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice8-7.png |

**4. Binding a template to a component**

|  |  |
| --- | --- |
| File | Description |
| Main.js | * Add the template directly to the templateContainer   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice8-8.png |
| MainController.js | * Call the update function of the templateContainer to update its content based on the data passed as parameter   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice8-9.png |

**5. Creating a dataview component**

|  |  |
| --- | --- |
| File | Description |
| Image.js | * Create a model called Image * Add the following fields:   + src string   + caption string * Remember to add the new model to the Application.js models collection   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice8-10.png |
| Images.js | * Create a store called Images with the Image model linked and some inline data * Remember to add the new store to the Application.js stores collection   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice8-11.png |
| MyFirstDataView.js | * Create a data view component with the Images store associated and a template to display the images * Remember to add the new view to the Application.js views collection   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice8-14.png |
| Main.js | * Add the MyFirstDataView component to the main container   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice8-13.png |
| MainController.js | * Add select listener to the dataview component and alert the selected image   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice8-15.png |

**Practice 9**

**1. Understanding components life cycle**

|  |  |
| --- | --- |
| File | Description |
| Main.js | * Remove the items property * Implement the initComponent function * Set the items property and add one container with itemId = mainContainerChild * After calling parent function, query the mainContainerChild and confirm that it is not ready yet. To do this print in the console the length of the result of the query   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice9-1.png   * Implement the afterRender function * After calling parent function, query the mainContainerChild and confirm that it is ready now. To do this print in the console the length of the result of the query   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice9-2.png   * Implement the destroy function * Add a defer function in the afterRender function. Wait for 2 seconds and then, remove all viewport childs. This way you will be able to see how the destroy function is called   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice9-3.png |

**2. Creating a custom field component**

|  |  |
| --- | --- |
| File | Description |
| ValidateImageField.js | * Create a new view called ValidateImageField * Extend from Ext.form.FieldContainer * Add the following items:   + xtype = image, itemId = imgToValidate   + xtype = checkboxfield, itemId = chxIsValid, boxLabel = Is Valid? * Add the view to the Application.js views collection * Add the view as a child of the mainContainer * Test the application   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice9-4.png |

**3. Making it a field like component**

|  |  |
| --- | --- |
| File | Description |
| ValidateImageField.js | * Add the following mixing:   + Ext.form.field.Field * Implement the initComponent function and initialize the mixins   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice9-5.png |

**4. Implementing the setValue function**

|  |  |
| --- | --- |
| File | Description |
| ValidateImageField.js | * Implement the setValue function and update the image and the checkbox with their corresponding values   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice9-6.png   * If you run the application you will see something like the following.  The component is not being rendered well.  To fix this we will need to detect the moment in which the image loading is complete and then, update the layout of the parent container.   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice9-7.png |

**5. Fixing the layout issue**

|  |  |
| --- | --- |
| File | Description |
| ValidateImageField.js | * Implement the afterRender function and call a function in which we will add the listeners we need to the image component * In the image load handler, call the doLayout function to refresh the layout of the component   D:\Documentos\Dropbox\Documentos\Personal\Software\Trainings\ExtJS 4 Basic to Advanced\Resources\Practices\practice9-8.png |

**6. Pendings**

1. Add a textfield not editable to the component in which the url will be displayed
2. Make the textfield editable, listen the blur event and then update the image src property
3. Update the url in the getValue function
4. If an invalid image was set, set the image with the default one
5. Implement the isValid function, the component will be invalid if there is no url typed or if the url is invalid

**Practice 10**

Run the “Sencha app build testing” and “Sencha app build production” commands on the Practice10 folder, then, review and test the builds.