Building RIA with Ext JS

by Akash Saxena

Who's your host anyway?

Akash

CTO, GetSetGo Fitness

And what is a RIA?

RICH INTERNET APPLICATION

Let's look at an example ...

Complexity and scaling

Database connectivity

Security and reliability

Enterprise Application

Focus areas

BEFORE MOVING AHEAD ...

... let's discuss some core concepts

This

Object literals

Javascript first class functions

Classes

APPLICATION OUTLINE

A standard layout that keeps the code clean

Module locations

Naming conventions

Coding conventions and house styles

Remember - other developers too are working on your project

SENCHA CMD

A super-utlity tool

Generator

Build service

Scaffolding tools

Rapid application development tools

APPLICATION ARCHITECTURE

Unified folder structure

Namespace:
<AppName>.<foldername>.
<ClassAndFileName>

Check the error if the file is not found

Let's look at boostrapping - index.html, app.js, application.js

Notice mainView

How does onAppUpdate work?

Let's look at Model/store and create user model/store

DESIGNING FOR DIFFERENT SCREENS

Classic
The classic toolkit provides traditional Ext JS 5 application support. This includes support for the desktop browsers, tablets, and touchscreen enabled laptops.

Modern
The modern toolkit provides universal HTML5 application support all modern browsers (IE11+) from desktop to phone.

Let's look at app.json and build

Let's look at the folder structure and watch classic or modern

The core concepts

Let's write ...

OUR FIRST EXTJS PROGRAM

THE CLASS SYSTEM

Let's write our first class

Let's look at Ext.classManager, Ext.base

Ext.define:
This shorthand is used to create a new class, extend a class, or whenever we need to apply some override(s) in a class.

Ext.create:
This shorthand creates a new instance of a class, using either the fullname class, the alias class, or the alternate name class. Using any of these options, the class manager handles the correct mapping to create the class. We can also use this shorthand to create objects from an existing class.

Ext.widget:
This shorthand is used to create a widget using the xtype (alias) property or a configuration object.

INHERITANCE

Let's check a simple inheritance

The supervisor class

Mixins - Mixing the classes

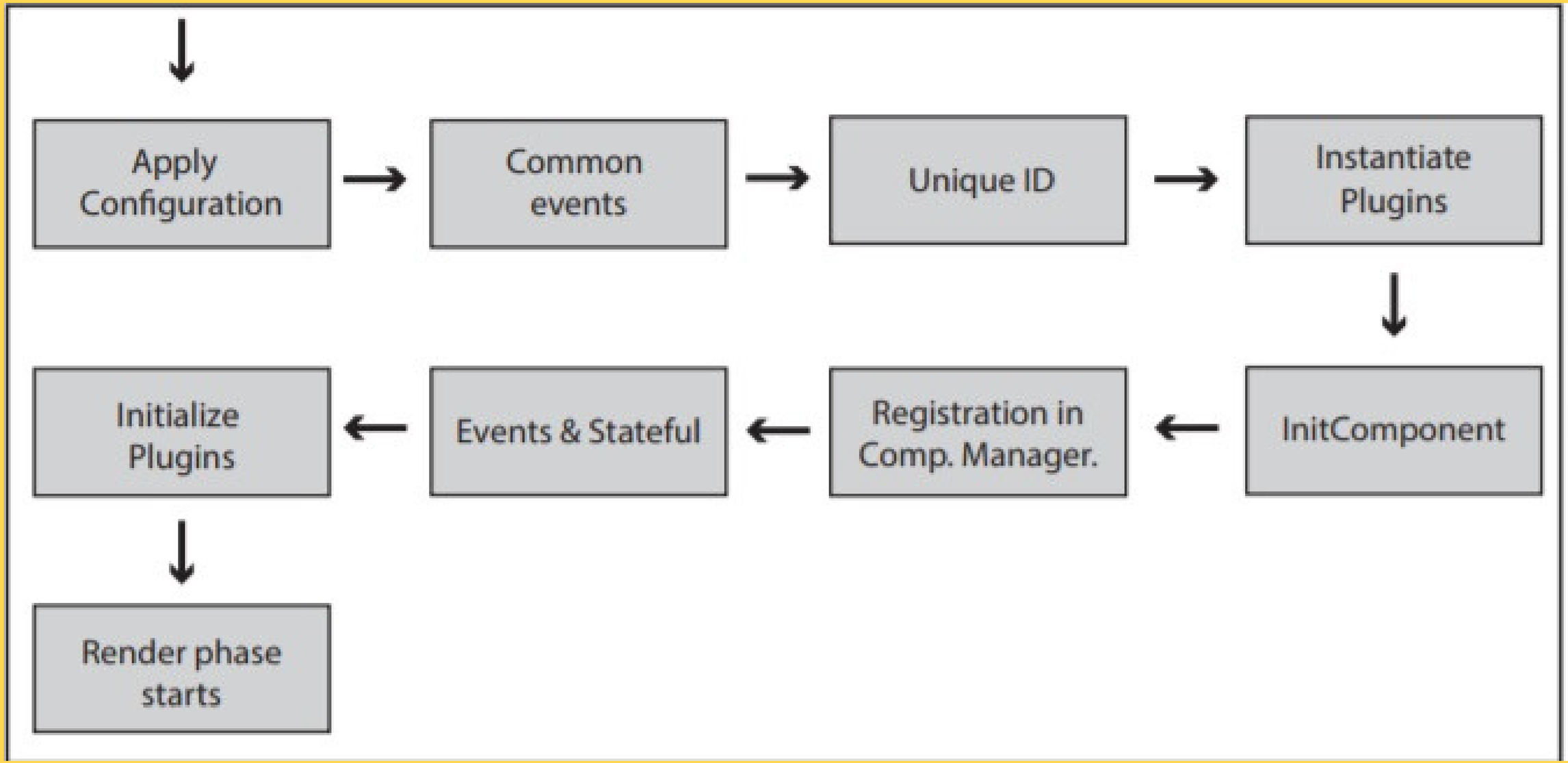
Let's look at the initConfig and setter and getter functions

Let's look at alias

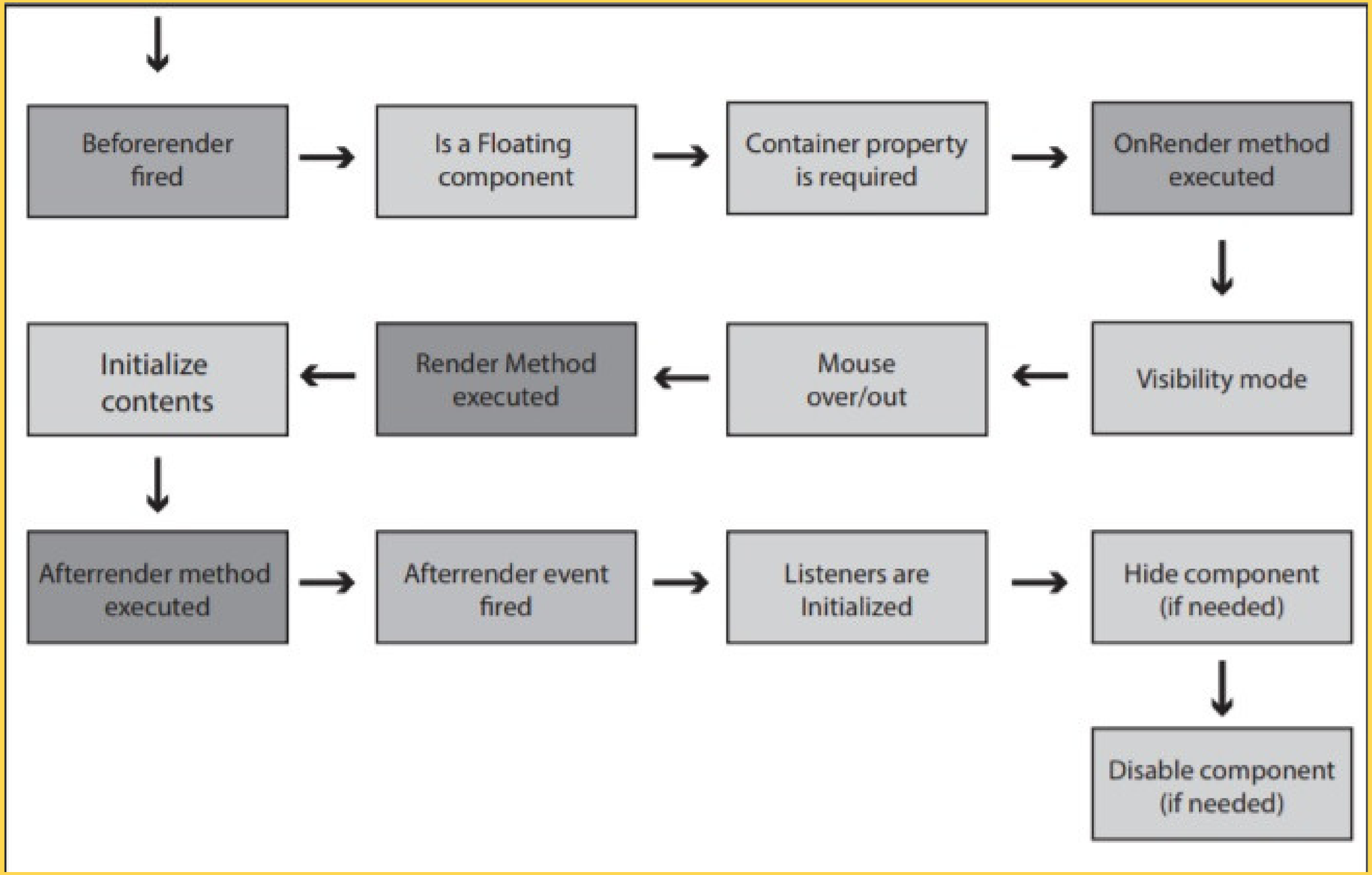
The component lifecycle

INITIALIZATION > RENDERING > DESTRUCTION

INITIALIZATION



RENDERING

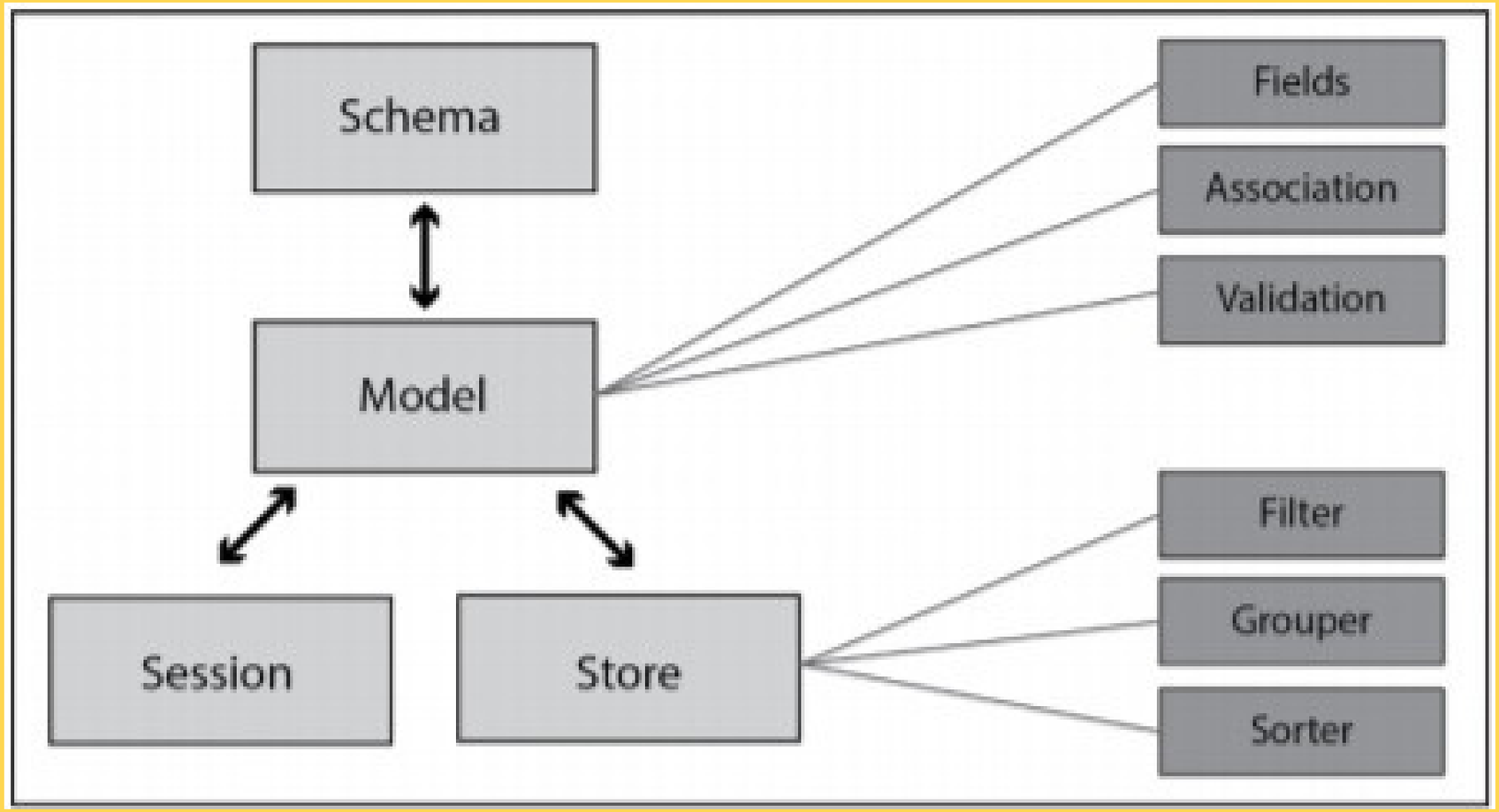

Beforerender fired
Is a Floating component
Container property is required
OnRender method executed
Visibility mode
Mouse over/out
Render Method executed
Initialize contents
Afterrender method executed
Afterrender event fired
Listeners are Initialized
Hide component (if needed)
Disable component (if needed)

Containers

LET'S WRITE A CONTAINER

Container	Description
`Ext.panel.Panel`	This component extends `Ext.container.Container` and is a container with specific functionality. It is also one of the most common containers used in Ext JS.
`Ext.window Window`	This component extends the `Ext.panel.Panel` class and is intended to be used as an application window. Windows are floating components and can be resized and dragged. Also, windows can be maximized to fill the viewport.
`Ext.tab.Panel`	This component also extends the `Ext.panel.Panel` class container and has the ability to contain other `Ext.panel.Panel` components, creating one tab per panel in its header section. Also, the tab panel uses the card layout to manage its child components.
`Ext.form.Panel`	The form panel extends the `Ext.panel.Panel` class and provides a standard container for forms. Essentially, it is a `Panel` container that creates a basic form for managing field components.
`Ext.Viewport`	This container represents the application area (browser viewport). It renders itself to the document body and resizes itself to the size of the browser viewport.

DATA MODELS


Schema
Model
Session
Store
Fields
Association
Validation
Filter
Grouper
Sorter

DATA MODELS

AJAX calls

Data model

Data model - setter and getter

One-to-one relationship

One-to-many relationship

MVC & MVVM

MVC

Model, View, Controller

MVVM

View-Model - The ViewModel is a class that manages data specific to the View.
It allows interested components to bind to it and be updated whenever this data changes.

VIEW-CONTROLLER

Listeners - Notice how the scope is resolved

Let's change the scope of the controller