Nexus Core 3.0

Simple is better

Author: Mike Zhang

Email: [winsource6@hotmail.com](mailto:winsource6@hotmail.com)

Create Date: 05/08/2014

Last Update Date: 25/08/2014

Nexus Core 3.0

# Life cycle

## Request Handler

* Module Handle
  + Domain check
  + Redirect to main domain from subdomain

## Global Application Start

* Nexus Engine Startup
  + Dependency engine setup (AutoFac)
  + Cache engine setup (not decide yet, use normal memory for now)
  + Initial setup
* Model Binder (Consider)
  + Model Metadata Provider (Consider)
* Custom View Engine
  + Add custom view template path
* Route Builder
  + Add Route Constrains

## Base Controller

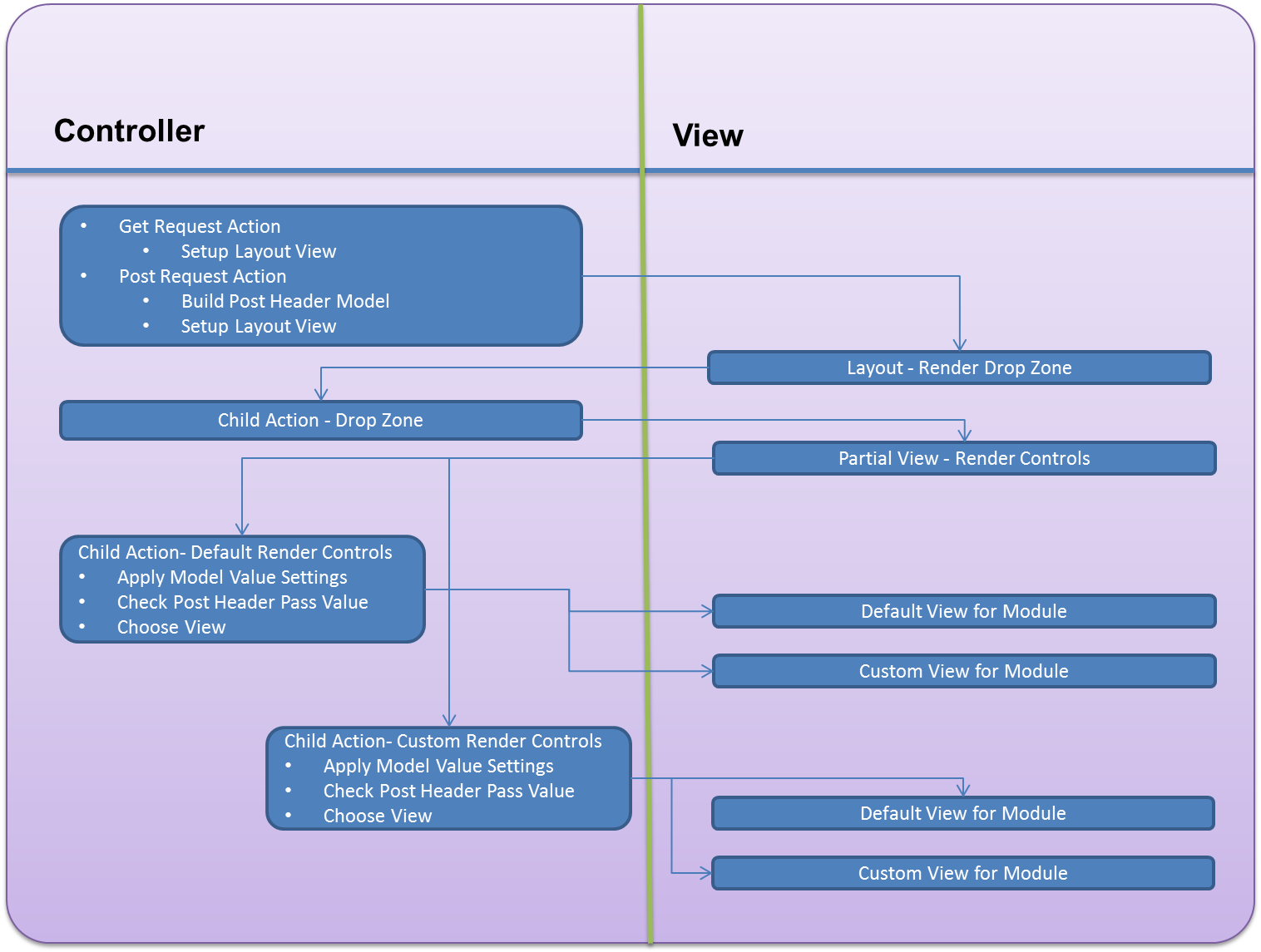
* Engine Context
  + Site Info Builder
  + Page Info Builder
  + Required JavaScript Builder
  + Required Css Builder
  + Page Module Model Builder
  + 301 handler
  + 404 handler

## Controller and View

* Controller
  + Get Request Action
  + Post Request Action
  + Error Request Action
  + Authentication Login Action
  + Default Drop Zone Child Action
  + Default Render Module Child Action
* View
  + Load Layout
  + Loop Drop Zone
  + Loop Modules in Drop Zone
  + Partial View for module default display
* Custom View
  + Partial View for module custom display

## Client Side JavaScript

* Default JavaScript
  + Bootstrap
  + Angular
  + Angular Master Controller
  + Http Ajax post
* Custom JavaScript
  + Depend on page



# Page Layout and Inherit

## Create

* A page can be create
* A page must have a parent node
* A page can inherit from layout
* A page can inherit from page
* A page cannot inherit from their own child page
* A page cannot inherit from their own child page in chain

### Modify

* A page can change parent to layout
* A page can change parent to page
* A page cannot change parent to their own child page
* A page cannot change parent to their own child page in chain

### Delete

* A page can be deleted
* A page cannot be deleted when it has child page
* A page cannot be deleted when it has been used in web node

# Business Logic Layer Service

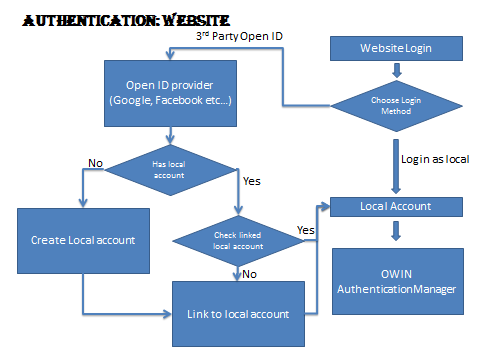
## Data Service Types

* Primitive: CRUD service for one entity only
* Composite: Composite service has join with multi entities, can see all Primitive Service
* Component : Component service with model container from both composite and primitive

# User membership and Authentication

## Website Login

Owin Authentication Manager will be used to manage for local user identity. Any other open ID provider will linked to a local account.



## Mobile App Login

Mobile App login will able to access via Web API service only, two authorization method will be used in Service. Application Token and User login Token, The final decision haven’t been made yet.

