NGPS Applications – Core of extensibility and configurability of the system

Every applications is identified by a unique name, for example “text\_editor” and is contained within a folder with the same name. Applications are found in the “plugins” folder:

./scripts

plugins

application\_name

main.js [ this is the application’s entry point, it is responsible for including dependencies and setting up the app]

Application structure:

main.js

resources – folder with graphics and other resources (sound, data, etc)

icon.png – application icon

To load a new app into the NGPS system

*loadAppCode(“name”, new app()); - this function loads an app with the given name*

How the app should be structured:

*var app = function(params){*

*this.config = {}; //it needs to have a config property*

*}*

1. The config object

***interface***: [ none | standard ] – none means the application is a servicing app and does not require the user to run or suspend it, standard: an application occupying a container that requires input to be redirected to it and therefore allows the user to enter it by clicking on a cover container ( invisible ) and exit it by clicking on an exit container

***cove****r*: a cover container used to overlap on the app while inactive, only for app with interface

***exit***: an exit container that is used to exit the app active mode, only for app with interface

2. What does an app object inherit?

The app object has a parent property, this parent contains system information about the app such as: ***appName, appPath, appFullPath***

these are useful when the application has subcomponents that need to be loaded. It needs to know the path to load them from.

To access these from the app’s code example

*var name = this.parent.appName;*

3. Save & Load operation.

Apps must explicitly request not to be saved by setting the save permission to false (this needs to be done for any container created by the app that must not be saved). This is generally for editor only apps or for global app that will be loaded anyway.

Some apps may require data to be saved so that when they are loaded in presentation mode the data will be available. Also certain apps require parameters to be passed to them when loaded on a container.

Both these options are available through the ***\_store*** object.

*store: {  
 startup*Args: {

*edit:{}, - startup arguments for when app starts in editor mode*

*present:{} – startup args. for when app starts in presentation mode*

*},*

*data:{} – an object containing any data the application needs saved*

}