NGPS Core Documentation

1. Structure

1. Folder structure

..

scripts [ contains the core of the engine ]

plugins [ contains built-in and purchased apps ]

style [ contains global styling mechanisms, Bootstrap ]

statistics [ contains statistic information collected from the user in order to improve the experience ]

index.html [ application entry point ]

1.1 Core code structure

1.1.0 Bare core

TweenMax.min.js [ awesome 2D transforms library ]

require.js [ used for dynamically loading js files]

drivers.js [generic utility and platform specific functions]

container.js [ container code, building block upon which the system is built]

interaction.js [ extension module that adds interaction possibility to a container ]

camera.js [ extension module that turns a container into a camera]

app.js [ extension module that allows a container to host an app ]

1.1.1 Factory Wrapper

factory.js [ does initial setup, links theme with container creation, is an adapter for container & camera creation ]

init [ initial setup function, creates the factory.root object which is the root of the presetation ]

setup [ function defined by external sript for custom initial setup ]

AMS [ Anti Monotony Script dictates how every container will be styled ]

. constructors [ folder containing all the custom setup scripts ]

. themes [ folder containing all the AMS’ ]

. descriptors

containers.js [ file containing all the container styles owned by the user ]

links.js [ file containing all the link styles owned by the user ]

1.1.2 Events

gem.js [ is a generic event manager that handles event registering, dispatching, unregistering and monitoring ]

1.1.3 Saving & Loading a presentation

save.js [ handles saving the presentation in a HTML format ]

load.js [ loads the presentation from a HTML format ]

1.1.4 Applications

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:

plugins

text\_editor

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

1.1.5 CLI

cli.js [ is a command line interface that allows NGPS developers to easily test and manipulate the system ]

1. Meet the APIs

\* Method parameters enclosed in < > are optional

2.1.0 container : building block of NGPS system ( everything lives inside a container )

Properties

|  |  |
| --- | --- |
| UID |  |
| DOMreference |  |
| parent |  |
| angle |  |
| scaleX |  |
| scaleY |  |
| properties |  |
| isLeaf |  |
| isApp |  |
| child |  |
| children |  |
| outgoing |  |
| incoming |  |
| events |  |
| allowMove |  |
| allowTrigger |  |

Callbacks

|  |  |
| --- | --- |
| onMouseDown |  |
| onMouseUp |  |
| onMoved |  |
| onTrigger |  |

Methods

|  |  |
| --- | --- |
| load(<parent>) |  |
| extend( extensions ) |  |
| strip ( extensions ) |  |
| addChild( properties ) |  |
| removeChild( UID ) |  |
| discard( ) |  |
| changeParent( parent ) |  |
| addPrimitive( descriptor ) |  |
| removePrimitive( ) |  |
| show( ) |  |
| hide( ) |  |
| getPos( cx, cy, <refX>, <refY> ) |  |
| getWidth( ) |  |
| getHeight( ) |  |
| getPureWidth( ) |  |
| getPureHeight( ) |  |
| getCenter( ) |  |
| getLocalPos( ) |  |
| setWidth( width ) |  |
| setHeight( height ) |  |
| setAngle( angle, <ox>, <oy> ) |  |
| putAt( x, y, <refX>, <refY> ) |  |
| move( dx, dy, <noevent> ) |  |
| scale( amount, <ox>, <oy>) |  |
| enlarge( amount ) |  |
| rotate( delta\_angle, <ox>, <oy> ) |  |
| getAncestors( node ) |  |
| greatestCommonParent( target ) |  |
| link( target, descriptor ) |  |
| unlink( target ) |  |
| unlinkAll() |  |
| changeLinkTarget( oldTarget, newTarget) |  |
| maintainLink( target ) |  |
| maintainLinks() |  |
| addEventListener(event, handler, <context>) |  |
| removeEventListener( event, handler, <context>) |  |
| loadApp(appName) |  |

Events

|  |  |
| --- | --- |
| loadContainer |  |
| addChild |  |
| removeChild |  |
| discardContainer |  |
| hideContainer |  |
| showContainer |  |
| changeParent |  |
| changeWidth |  |
| changeHeight |  |
| changeAngle |  |
| changePosition |  |
| link |  |
| unlink |  |
| linkChange |  |
| appLoaded |  |

* + 1. Interact
    2. Camera
    3. App

* 1. Factory

Properties

|  |  |
| --- | --- |
| initialized | 0 if the factory is not initialized  1 if the factory is initialised |
| defaultContainer | a default container descriptor |
| settings | an object containing factory settings |
| root | the root container of the presentation |
| AMS | the Anti Monotony Script. Loaded from /scripts/themes/ |

Methods

|  |  |
| --- | --- |
| init( ) | initializes the factory, configures the factory settings and runs the custom setup function is it is defined. |
| setup( ) | custom function defined by any script in /scripts/constructors/  used to create custom setups ( presentation themes ) |
| newContainer( possize, tag, <parent>) | creates a new container.  possize: specifies the position and size of the container ( format {x:0,y:0,width:100,height:100} )  tag: the name of a descriptor to used in order to style the container  parent: the parent container where the new container will be attached |
| createContainer( descriptor, <parent>) | creates a new container entirely described by the descriptor passed  parent: specifies the container parent where the new container will be attached |
| newCamera( possize, tag, <parent>, <interval>) | creates a new camera:  possize: same as for container  tag: same as for container  parent: same as for container  interval: the period between camera operation ticks |
| newGlobalApp( appName ) | loads a new global application, it adds the app containers as children of the factory.root |

2.2.1 Descriptors

2.2.2 Anti Monotony Scripts

* 1. General Events Manager (GEM)

Properties

|  |  |
| --- | --- |
| events | Stores all event handlers and their required information ( object, run\_context ) |
| debug | flag that determines if GEM prints events on the CLI.  Default: false |

Methods

|  |  |
| --- | --- |
| fireEvent( data ) | fires a GEM event |
| addEventListener( event, object, handler, <run\_context> ) | adds an GEM event listener.  Parameters:  event: name of the event [string]  object: the object on which the event occurs [object]  handler: function that handles the event  run\_context: object that needs to be accessible from inside the handler[object]  \* if run context is present, handler needs to be a string naming a member function of run\_context |
| removeEventListener( event, object, handler) | removes event listener |
| cancelAll( ) | cancels all event listeners |
| list( verbose ) | lists all the GEM event listeners |

2.4 Apps

An apps main.js file contains all the necessary code to include dependencies, setup the app and comply with the NGPS environment. This file has a set structure so that apps can be easily loaded and managed by NGPS, therefore every app is a class ( object ) with the following structure:

var myApp = function( data )

{

this.config = { <add configuration properties here> };

this.parent = data['parent'];

this.init = function(){ } //called only one when bound to container

this.run = function(){ } //called whenever the container is triggered

this.suspend = function(){ } //called whenever the container looses focus ( or gets out of view )

this.shutdown = function(){ }//called only when app is unloaded from container

this.show = function(){ } //shows app

this.hide = function(){ } //hides app

}

//this function must be called in order to load the app into NGPS

loadAppCode("myAppName",myApp);

NGPS apps have complete control over events, therefore if an app is loaded in a container then it becomes unmovable. To solve this NGPS has a built in system that gives or takes the app control over events. By clicking once on a container with an app it will cause the app to run ( the run function is called ) and therefore all events will be diverted to the app. A round tick will be displayed in the upper left corner which if clicked will suspend the app ( the suspend function is called ) and take back event control from the app making the div movable again.

* 1. Command Line Interface (CLI)

\* Used as a developer tool

Properties

|  |  |
| --- | --- |
| node | the NGPS container that the console is focused on ( in order to operate on containers the CLI needs to focus on a container and then it can call the container’s member functions or reference it in other functions ) |
| UI | CLI UI div reference |
| UI.UIout | CLI output div reference |
| UI.UIin | CLI input field reference |

Methods

|  |  |
| --- | --- |
| showPrompt( <message> ) | shows the NGPS CLI prompt. Optional parameter message is a string and is appended to the prompt string. |
| show( ) | shows the CLI |
| hide( ) | hides the CLI |
| keysHandler( e ) | handles keyboard events and executs command when enter key is pressed |
| onExec( e ) | executes command in UI.UIin and updates the output |
| fetchParameters( parameters ) | evaluates parameters ( the raw string command must be split into separate strings separated by white spaces and passed to this function) |
| execute( command\_string ) | executes command passed as a string |
| shobjects( object ) | shows the members of an object |
| help( ) | displays information about the CLI on the NGPS CLI prompt |
| shtree( ) | shows the NGPS container tree |
| sh( ) | shows information about the current container the CLI is focusd on |
| cn( id ) | changes the CLI node to the container with the passed id. ( Focuses the CLI on a new node ) |
| rst( ) | resets the CLI node to the factory.root |
| ldtest( test\_name ) | loads a test script from /scripts/tests/ |
| require( script ) | includes a js |
| debugCondig( ) | configures the NGPS engine for debugging and testing  (it initialized the factory, loads a FPS meter and loads the benchmark test suite ) |