2

-
-
  - startstop
  -
  -
    - 1.
    - 2.
    - 3.
    - 4.

(load)	PluginContext ClassPath Class
(run)	PluginClassLoader ExtraClassManager
stop	ExtraClassManager PluginContext PluginClassLoader
forbid	(stop) plugin.xmlactive
enable	plugin.xmlactive run
unload	load stop PluginContextClass PluginContext
install	Home
uninstall	

update	stop

startstop

startstop

9.0stoplifecycle-monitor

lifecycle-monitor

```
<lifecycle-monitor class="com.fr.plugin.performance.base.manager.monitor.PerformancePluginMonitor"/>
```

AbstractPluginLifecycleMonitor

```
/**
 * ,
 */
public class PerformancePluginMonitor extends AbstractPluginLifecycleMonitor {

    @Override
    public void afterRun(PluginContext context) {
        //do something
    }

    @Override
    public void beforeStop(PluginContext context) {
       //release resources
    }
}
```

afterRunbeforeStopPluginContext

2017126

lifecycle-monitorafterInstall()beforeUninstall()afterUpdate()

`AbstractPluginLifecycleMonitor`

ExtraClassManager

stop

1.

PluginEventListenervoid on(PluginEvent)

1. priority,
2. scopeServletPluginListenerScope.ServletContextScopeServlet

```
//
PluginEventListener listener = new PluginEventListener(PluginListenerPriority.StableFactory,
PluginListenerScope.ServletContextScope) {

    @Override
    public void on(PluginEvent event) {

        refreshJavaScriptFiles();
    }
};
```

2.

PluginFilterPluginContextPluginFilterPluginEventListener

```
PluginFilter filter = new PluginFilter() {

    @Override
    public boolean accept(PluginContext context) {
        //JavaScriptFileHandler
        return context.contain("JavaScriptFileHandler");
    }
};
```

3.

PluginEventType

4.

GeneralContextPluginListenerRegistration

```
//
PluginEventListener listener = new PluginEventListener(PluginListenerPriority.StableFactory,
PluginListenerScope.ServletContextScope) {

    @Override
    public void on(PluginEvent event) {

        refreshJavaScriptFiles();
    }
};
//
PluginFilter filter = new PluginFilter() {

    @Override
    public boolean accept(PluginContext context) {
        //JavaScriptFileHandler
        return context.contain("JavaScriptFileHandler");
    }
};
//JavaScriptFileHandlerjs
PluginListenerRegistration.getInstance().listen(PluginEventType.AfterStop,listener,filter);
```