Design: Container

Purpose

The purpose of this document is to capture designs and considerations for building the container for the Paragon Desktop product space.

Goals

- The main purpose of the container is to host one (and only one) Paragon Application
- The container loads and "runs" a single paragon application specified on the command line

Functional Specification

Stories

Epic	Group	Story
Application Launch		• Start a paragon application based on command line arguments • Command line arguments should be as required by DA/Launcher (a.k.a. Life Cycle Management Plugin)
Process Management	Application Start	• Container should load the kernel plugins and application plugins. • If the application is a packaged application, follow appropriate start-up sequence as defined in the paragon.app.runtime plugin (use event page). • If the application is a hosted application, follow the start-up sequence of legacy paragon shell application (call the main function). • If the application is a JSI application, first launch the JSI process and follow the start-up sequence defined in the paragon.app.runtime

	Application stop	• Stop the current paragon application gracefully • The exact mechanism should be decided in conjunction with the design of (App Launcher, a.k.a. Life Cycle Management Plugin of) DA/Launcher.
	Application restart	• If the application is re-started (not loaded for the first time) it must notify the container • This aspect should be designed in conjunction with the design of workspace management
Window Management	Docking	• Window docking to screen edges (app bar)
	Snapping	• Windows should be able to snap to other windows
	Window types	• The container should support standard Chrome windows (as defined by the UX team) with configurable optional adornments • The container should support frameless (non-Chrome) windows

Key Concepts

Paragon container/host

Paragon container/host is responsible for running one and only one paragon application whose identity is passed in as command line argument.

Types of applications

There are essentially three kinds of paragon applications.

Packaged Application is an application whose markup (HTML, CSS etc.), resources (images, fonts etc.) code (javascript) and metadata (manifest) are compressed into a self-contained, signed archive.

A **hosted application** is an application whose markup, resources code are hosted on a server, while the metadata is (optionally) packaged into signed archive.

A **JSI application** is a kind of hosted application that has an associated JSI process. The hosted application's startup URL is provided by the JSI process upon successful launch.

Application manifest

An application manifest is a JSON file consisting of metadata of an application. The structure of the manifest is based on that of chrome applications. Both hosted and packaged application need an application manifest. The manifest of a packaged application must contain background script(s) that create application windows and monitor application life-cycle. The manifest of a hosted application must contain launch information such as start-up URL of the application and the bounds of the main application window. Paragon container constructs an manifest for a hosted application that does not have one.

Event page

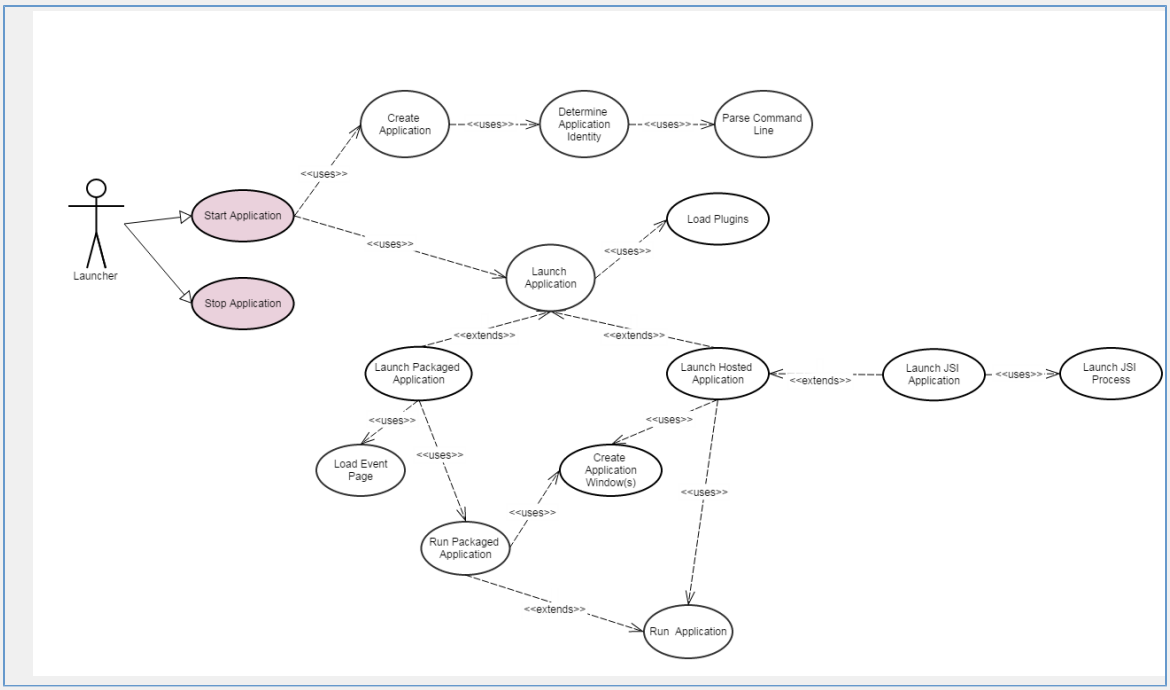
An event page is a page constructed out of the background script(s) of a packaged application. Event page essentially bootstraps a packaged application by creating and showing one or more application windows. Event page is loaded by the container as and when needed. If the background scripts do not create any application windows within a given amount of time the application is terminated.

Application window

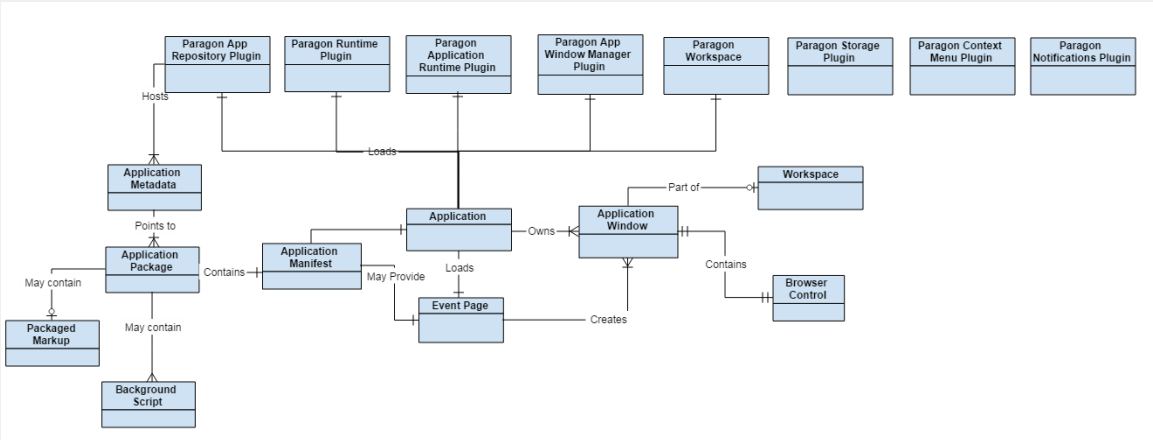
A paragon application window consists of an optional frame with its adornments (title bar, border, icon, system buttons, re-size handles etc.) and a single instance of a paragon browser control. Application windows can be created by executing **paragon.app.window.create**() or **window.open**() API. A packaged application creates it's initial window(s) in the background scripts of the event page, where as a hosted application depends of the host to create its main window.

Design

Use Cases

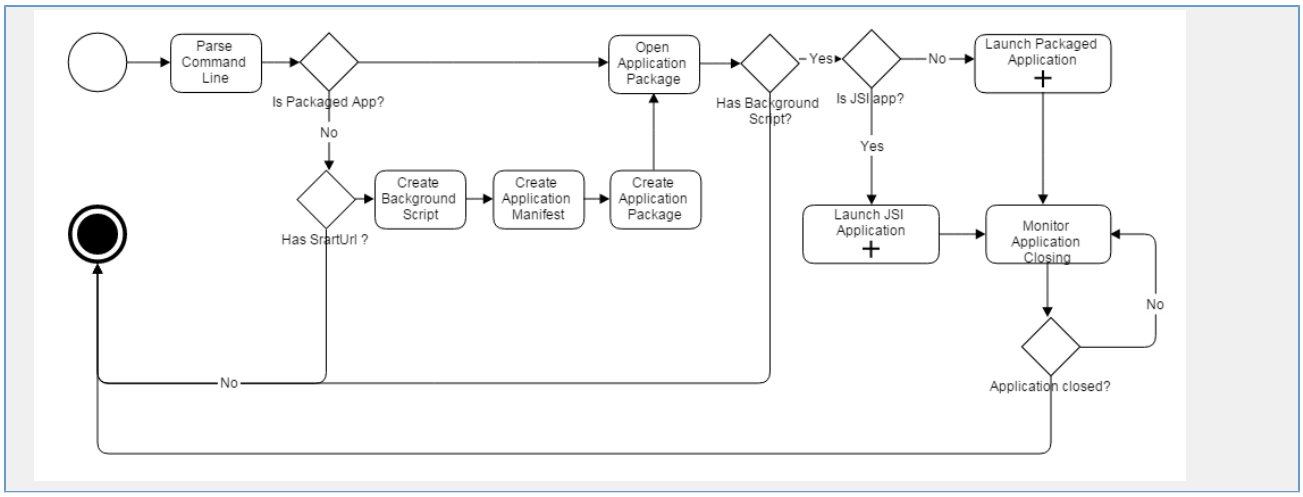


High level (logical)

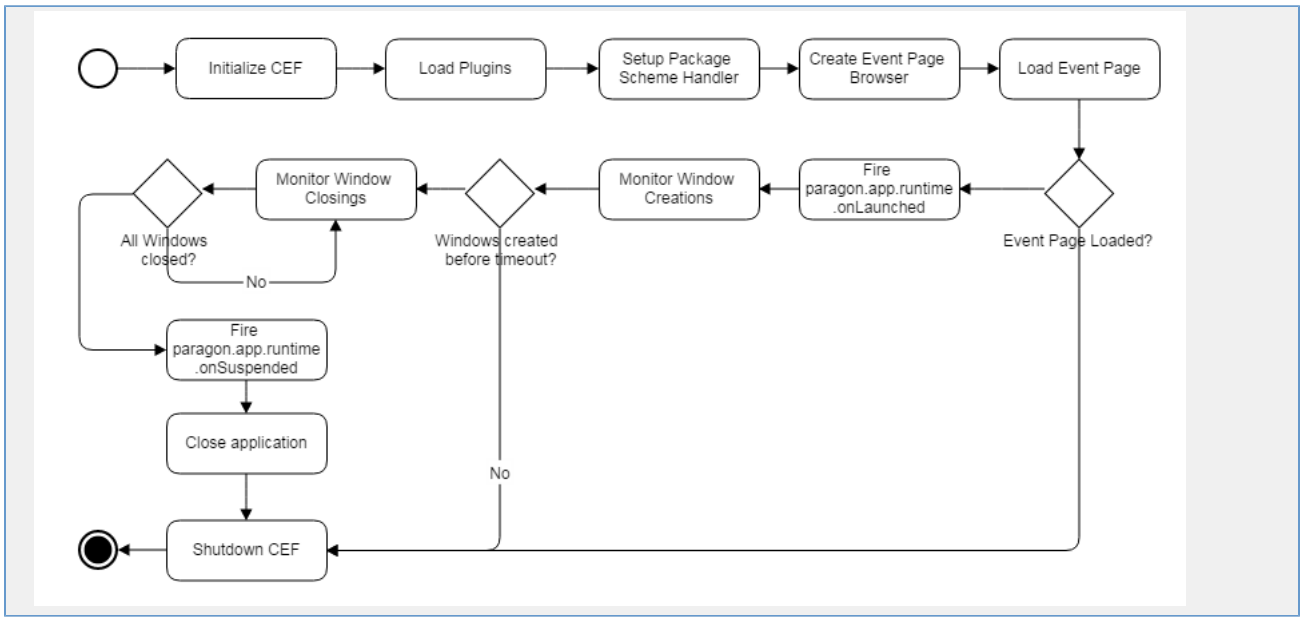


Paragon container flows

High Level



Packaged application launch



JSI application launch