Hands-On Lab

Process Lifetime Management

Lab version: 1.0.0

Last updated: 6/11/2012



Contents

[Overview 3](#_Toc327187695)

Overview

Process Lifetime Management, or PLM, is one of the most important concepts for a developer building Metro style apps to understand. Unlike traditional Windows applications, which continue to execute even when they are in the background, Metro style apps only execute when they are in the foreground. (Note that snapped apps still count as foreground.) In general, an app that is not visible to the user is suspended by the operating system and unable to execute until the OS restores it to the foreground.

When an application is suspended, it remains in memory with all its threads suspended. As long as the process remains in memory, the application resumes executing where it left off when it comes back to the foreground and there is nothing you, the developer, have to do to make that happen. However, you are not *guaranteed* that the process will remain in memory. If, while a Metro app is suspended, Windows determines that it needs the memory the suspended app is consuming, Windows can *terminate* the suspended app. If that happens, all state that has not been saved is lost. Unless you take steps to preserve that state, the user will be in for a surprise when he or she swaps back to the application and finds that all the work he or she has done is gone.

Obviously, we cannot allow this to happen. You never know whether a suspended application is going to be terminated. At the time the application is suspended, even Windows doesn’t know whether the application will eventually be terminated, You should assume that your app will be terminated and you must write code to save the state of the app when it is suspended and restore it upon reactivation if you detect that a termination occurred while the app was suspended.

Sound tricky? It can be sometimes, but for most apps it is not and it is work you have to do to be a great app. The cost of not doing so is the potential for a user to lose everything he or she has done in *your a*pp simply by switching briefly to another application.

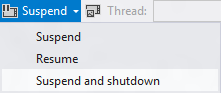
Because the only state that needs to be saved in Contoso Coookbook is the navigation state – the item or group the user was viewing and the navigation history – you don’t have to do anything to handle PLM. Visual Studio included a line of code in the app.oncheckpoint handler in default.js that serializes the navigation state into session state:

app.sessionState.history = nav.history;

In addition, Visual Studio included code in the activated event handler that restores the navigation state if the app was terminated by the operating system after it was suspended:

* 1. if (app.sessionState.history)
  2. {
  3. nav.history = app.sessionState.history;
  4. }

The result of all this is that you get a lot for free. If Contoso Cookbook is suspended and terminated, it automatically goes back to the last page you were viewing when it is restarted. You can test this by launching the app from Visual Studio with F5, selecting a recipe, and selecting Suspend and shutdown from the Debug Location toolbar:



After shutting the application down this way, press F5 to relaunch the app. This simulates what happens when the app is terminated and relaunched by the operating system. Thanks to Visual Studio, the app goes back to the recipe you were viewing when you shut it down. Because the navigation history was restored as well, you can even use the back button to retrace your steps through the app.