

Background Agents Debugging Tips on Windows Phone

This article lists few tips to test your background agents working.

Introduction



During development of scheduled tasks/background agents you can call `ScheduledActionService.LaunchForTest()` to allow you to run your agent more regularly than the 30 minutes allowed in production, and also call standard debug methods.

As a new Windows Phone developer, I really struggled with debugging background agents working with real 30 mins launch scenario. The following tips may help you with debugging - many are obvious, but they are also things that are easy to overlook.



Note: The tips and steps below have been tested for `PeriodicAgent` but not `ResourceIntensiveTask`.

Log to IsolatedStorage

The best way to log progress in a background agent is to write to your isolated storage.

You can read your isolated storage contents using the [Isolated Storage Tool](#) (only for Windows Phone 8).

Capture logs and view them in the app

You can also capture logs and view them in the app itself. This is particularly useful when debugging during development, but it can be useful after release in some circumstances (it is more usual to provide an interface to allow debug logs to be shared or emailed).



Note: For released apps, while it can be useful to have access to debugging information, it can be confusing to the user to have it visible. A good way to make debugging conditionally available to users is to make it available as an app preference or to provide a [URI Association](#) so you can provide a link to turn the debug page on when it is needed.

[File:ScheduledTaskLogger.zip](#) provides code to capture logs and view them in the app. The zip includes:

- **ScheduledTaskLogger.cs** - This logs data while the scheduled task is running. You can set how many logs to keep, and also whether it writes out to file after every log message - probably not a good idea for release, but useful if there's a problem that's causing a crash. I also try to save the log in `ScheduledAgent_UnhandledException`.
- **ScheduledTaskLog.xaml** and **ScheduledTaskLog.xaml.cs** - these are the views in the main app that you can use to view the logs. There's also a button to email a particular log to support.

At the beginning of your scheduled task add a call to create the logger as shown:

```
ScheduledTaskLogger _logger=new ScheduledTaskLogger();
_logger.Load();
_logger.NewEntry();}}
```

Add logging calls as shown below:

```
_logger.Log("text goes here");
```

In your "About" page (or similar), add a button to navigate to the **ScheduledTaskLog.xaml** page.

Log to toast notifications

You can put a `Toast` notification inside your background agent - this can be useful to provide visual indication when a background task has run/is running.

Note however that since toast notifications automatically disappear after few seconds and may be missed these should not be used for "critical" notification or logging.

Debug on Windows Phone 7

If possible, debug your background agents on Windows Phone 7. Debugging takes a lot of time - and Windows Phone 8 puts strict constraints on memory and time limitations for background agent even while debugging.

As there are no such constraints when debugging on Windows Phone 7 this gives you more time to play with.

Check you're using supported APIs

Some APIs cannot be used inside a background agent. Go through the list of unsupported APIs [here](#) 

Other tips

- Debug statements do not work when you are in release mode! Putting `Debug.WriteLine()` inside a `BackgroundAgent` is not a good way of testing. It works only when you invoke background agent using `ScheduledActionService.LaunchForTest()` but not when OS has scheduled it.
- Even if you add a `ScheduleAgent` by **New | Windows Phone Schedule Task Agent** make sure an "ExtendedTask" is added to your main projects **WMAAppManifest.xml**. If not, add it in **WMAAppManifest.xml** yourself using format below:

```
<Tasks>
  <DefaultTask Name="_default" NavigationPage="MainPage.xaml" />
  <ExtendedTask Name="BackgroundTask">
    <BackgroundServiceAgent Specifier="ScheduledTaskAgent" Name="Task_Name"
Source="Assembly_Name" Type="Assembly_Name.ScheduledAgent" />
  </ExtendedTask>
</Tasks>
```