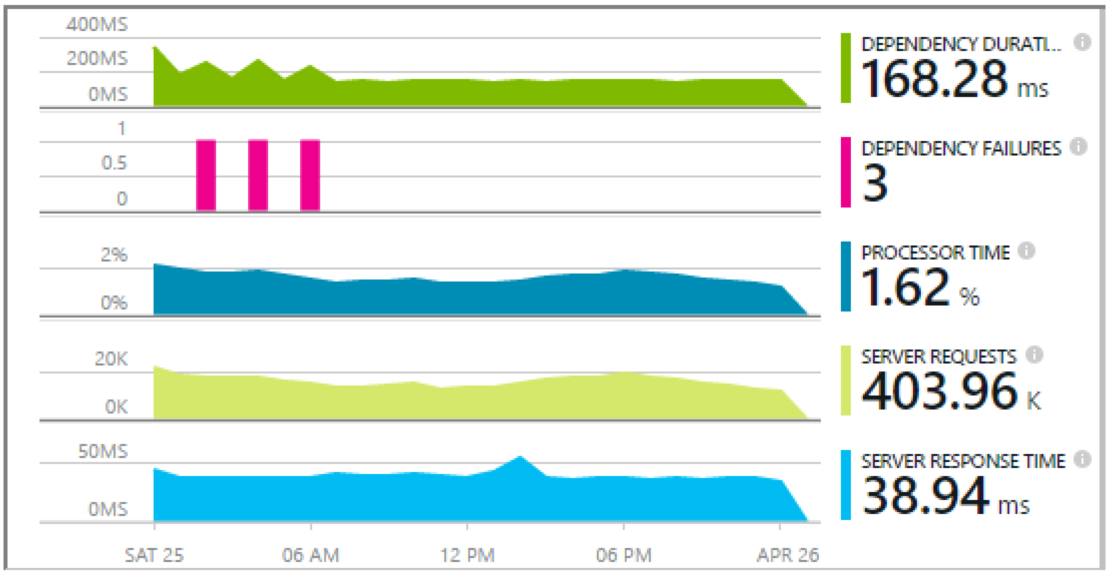
Instrument web apps at runtime with Application Insights

05/05/2017 • 7 minutes to read • Contributors all

In this article

You can instrument a live web app with Azure Application Insights, without having to modify or redeploy your code. If your apps are hosted by an on-premises IIS server, install Status Monitor. If they're Azure web apps or run in an Azure VM, you can switch on Application Insights monitoring from the Azure control panel. (There are also separate articles about instrumenting live J2EE web apps and Azure Cloud Services.) You need a Microsoft Azure subscription.



You have a choice of three routes to apply Application Insights to your .NET web applications:

- **Build time:** Add the Application Insights SDK to your web app code.

- **Run time:** Instrument your web app on the server, as described below, without rebuilding and redeploying the code.
- **Both:** Build the SDK into your web app code, and also apply the run-time extensions. Get the best of both options.

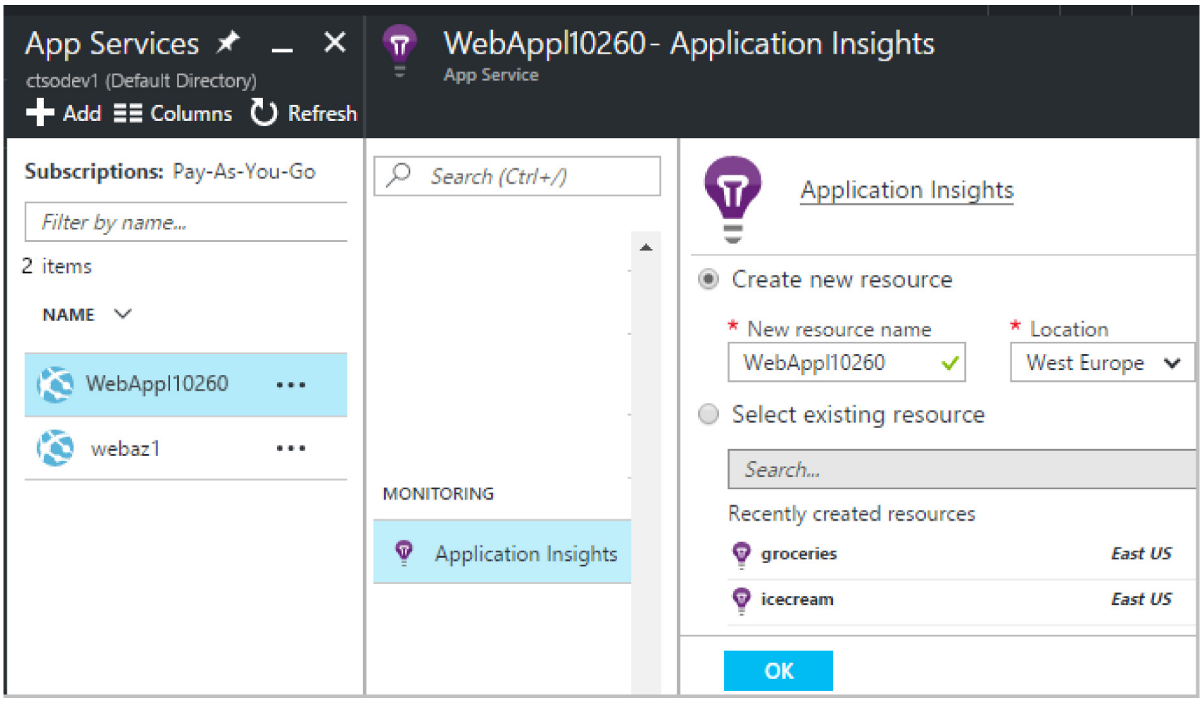
Here's a summary of what you get by each route:

	Build time	**Run time**
Requests & exceptions	Yes	Yes
More detailed exceptions		Yes
Dependency diagnostics	On .NET 4.6+, but less detail	Yes, full detail: result codes, SQL command text, HTTP verb
System performance counters	Yes	Yes
API for custom telemetry	Yes	No
Trace log integration	Yes	No
Page view & user data	Yes	No
Need to rebuild code	Yes	No

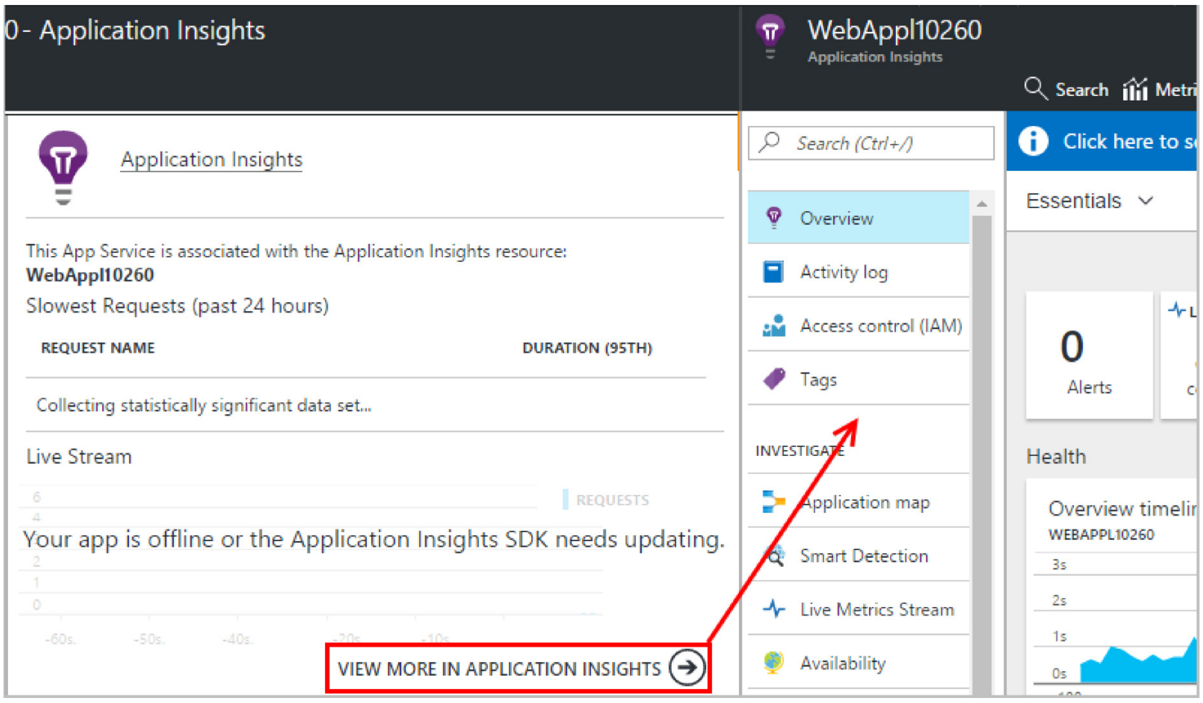
Monitor a live Azure web app

If your application is running as an Azure web service, here's how to switch on monitoring:

- Select Application Insights on the app's control panel in Azure.



- When the Application Insights summary page opens, click the link at the bottom to open the full Application Insights resource.



[Monitoring Cloud and VM apps](https://docs.microsoft.com).

Enable client-side monitoring in Azure

If you have enabled Application Insights in Azure, you can add page view and user telemetry.

1. Select Settings > Application Settings
2. Under App Settings, add a new key value pair:

Key: `APPINSIGHTS_JAVASCRIPT_ENABLED`

Value: `true`

3. **Save** the settings and **Restart** your app.

The Application Insights JavaScript SDK is now injected into each web page.

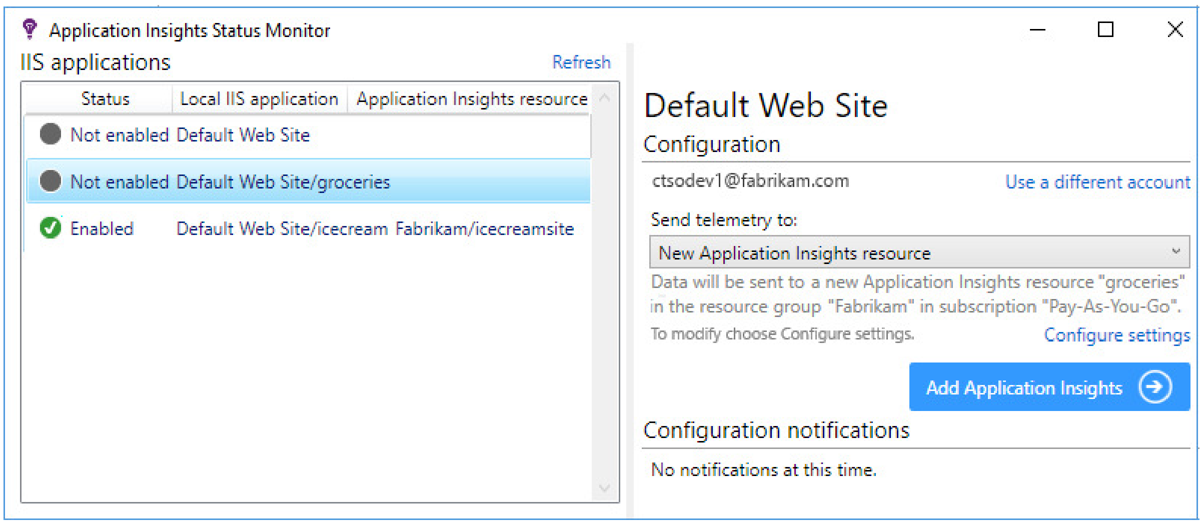
Monitor a live IIS web app

If your app is hosted on an IIS server, enable Application Insights by using Status Monitor.

1. On your IIS web server, sign in with administrator credentials. 1
2. If Application Insights Status Monitor is not already installed, download and run the Status Monitor installer (or run Web Platform Installer and search in it for Application Insights Status Monitor).

3. In Status Monitor, select the installed web application or website that you want to monitor. Sign in with your Azure credentials.

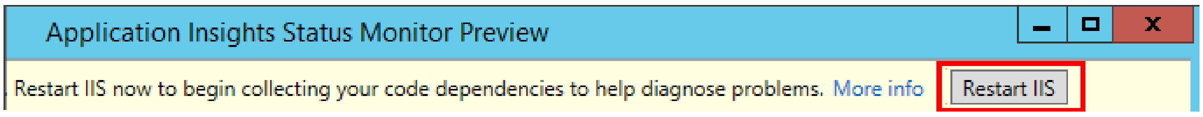
   Configure the resource where you want to see the results in the Application Insights portal. (Normally, it's best to create a new resource. Select an existing resource if you already have web tests or client monitoring for this app.)



Status	Local IIS application	Application Insights resource
● Not enabled	Default Web Site	
● Not enabled	Default Web Site/groceries	
✓ Enabled	Default Web Site/icecream	Fabrikam/icecreamsite

Default Web Site
Configuration
ctsodev1@fabrikam.com
Send telemetry to:
New Application Insights resource
Data will be sent to a new Application Insights resource "groceries" in the resource group "Fabrikam" in subscription "Pay-As-You-Go".
To modify choose Configure settings.
Configuration notifications
No notifications at this time.

4. Restart IIS.



Restart IIS now to begin collecting your code dependencies to help diagnose problems.

Your web service is interrupted for a short while.

Customize monitoring options

Enabling Application Insights adds DLLs and ApplicationInsights.config to your web app. You can edit the .config file to change some of the options.

When you re-publish your app, re-enable Application Insights

Before you re-publish your app, consider adding Application Insights to the code in Visual Studio. You'll get more detailed telemetry and the ability to write custom telemetry.

If you want to re-publish without adding Application Insights to the code, be aware that the deployment process may delete the DLLs and ApplicationInsights.config from the published web site. Therefore:

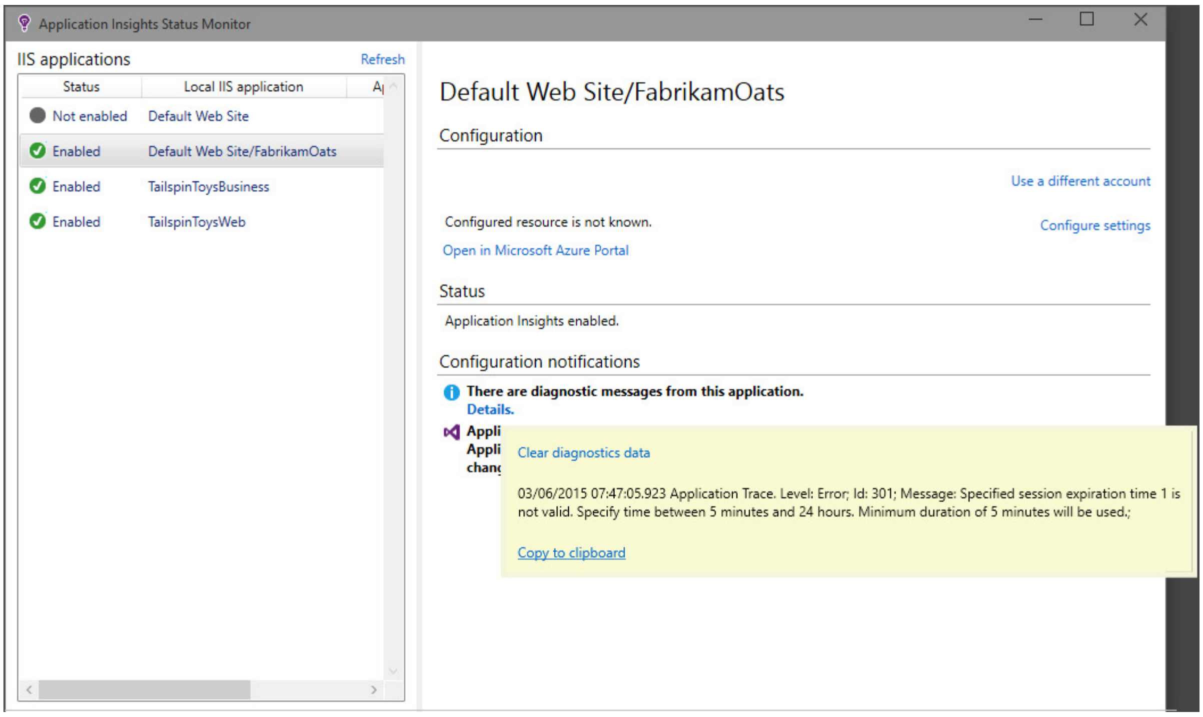
1. If you edited ApplicationInsights.config, take a copy of it before you re-publish your app.
2. Republish your app.
3. Re-enable Application Insights monitoring. (Use the appropriate method: either the Azure web app control panel, or the Status Monitor on an IIS host.)
4. Reinstate any edits you performed on the .config file.

Troubleshooting runtime configuration of Application Insights

Can't connect? No telemetry?

- Open the necessary outgoing ports in your server's firewall to allow Status Monitor to work.

- Open Status Monitor and select your application on left pane. Check if there are any diagnostics messages for this application in the "Configuration notifications" section:


Application Insights Status Monitor
IIS applications
Refresh
Status
Local IIS application
Not enabled Default Web Site
Enabled Default Web Site/FabrikamOats
Enabled TailspinToysBusiness
Enabled TailspinToysWeb
Default Web Site/FabrikamOats
Configuration
Use a different account
Configured resource is not known.
Configure settings
Open in Microsoft Azure Portal
Status
Application Insights enabled.
Configuration notifications
There are diagnostic messages from this application.
Details.
Clear diagnostics data
Copy to clipboard

03/06/2015 07:47:05.923 Application Trace. Level: Error; Id: 301; Message: Specified session expiration time 1 is not valid. Specify time between 5 minutes and 24 hours. Minimum duration of 5 minutes will be used.;

- On the server, if you see a message about "insufficient permissions", try the following:
  - In IIS Manager, select your application pool, open **Advanced Settings**, and under **Process Model** note the identity.
  - In Computer management control panel, add this identity to the Performance Monitor Users group.
- If you have MMA/SCOM (Systems Center Operations Manager) installed on your server, some versions can conflict. Uninstall both SCOM and Status Monitor, and re-install the latest versions.
- See Troubleshooting.

System Requirements

OS support for Application Insights Status Monitor on Server:

- Windows Server 2008
- Windows Server 2008 R2
- Windows Server 2012
- Windows server 2012 R2
- Windows Server 2016

with latest SP and .NET Framework 4.5

On the client side: Windows 7, 8, 8.1 and 10, again with .NET Framework 4.5

IIS support is: IIS 7, 7.5, 8, 8.5 (IIS is required)

Automation with PowerShell

You can start and stop monitoring by using PowerShell on your IIS server.

First import the Application Insights module:

```
Import-Module 'C:\Program Files\Microsoft Application Insights\Status
Monitor\PowerShell\Microsoft.Diagnostics.Agent.StatusMonitor.PowerShell.dll'
```

Find out which apps are being monitored:

```
Get-ApplicationInsightsMonitoringStatus [-Name appName]
```

- `-Name` (Optional) The name of a web app.
- Displays the Application Insights monitoring status for each web app (or the named app) in this IIS server.

- Returns `ApplicationInsightsApplication` for each app:
  - `SdkState==EnabledAfterDeployment`: App is being monitored, and was instrumented at run time, either by the Status Monitor tool, or by `Start-ApplicationInsightsMonitoring`.
  - `SdkState==Disabled`: The app is not instrumented for Application Insights. Either it was never instrumented, or run-time monitoring was disabled with the Status Monitor tool or with `Stop-ApplicationInsightsMonitoring`.
  - `SdkState==EnabledByCodeInstrumentation`: The app was instrumented by adding the SDK to the source code. Its SDK cannot be updated or stopped.
  - `SdkVersion` shows the version in use for monitoring this app.
  - `LatestAvailableSdkVersion`shows the version currently available on the NuGet gallery. To upgrade the app to this version, use `Update-ApplicationInsightsMonitoring`.

```
Start-ApplicationInsightsMonitoring -Name appName -InstrumentationKey 00000000-000-000-000-
0000000
```

- `-Name` The name of the app in IIS
- `-InstrumentationKey` The ikey of the Application Insights resource where you want the results to be displayed.

- This cmdlet only affects apps that are not already instrumented - that is, SdkState==NotInstrumented.

  The cmdlet does not affect an app that is already instrumented. It does not matter whether the app was instrumented at build time by adding the SDK to the code, or at run time by a previous use of this cmdlet.

The SDK version used to instrument the app is the version that was most recently downloaded to this server.

To download the latest version, use Update-ApplicationInsightsVersion.

- Returns `ApplicationInsightsApplication` on success. If it fails, it logs a trace to stderr.

```
Copy

    Name                       : Default Web Site/WebApp1
    InstrumentationKey         : 00000000-0000-0000-0000-000000000000
    ProfilerState              : ApplicationInsights
    SdkState                   : EnabledAfterDeployment
    SdkVersion                 : 1.2.1
    LatestAvailableSdkVersion : 1.2.3
```

`Stop-ApplicationInsightsMonitoring [-Name appName | -All]`

- `-Name` The name of an app in IIS
- `-All` Stops monitoring all apps in this IIS server for which `SdkState==EnabledAfterDeployment`
- Stops monitoring the specified apps and removes instrumentation. It only works for apps that have been instrumented at run-time using the Status Monitoring tool or Start-ApplicationInsightsApplication. (`SdkState==EnabledAfterDeployment`)
- Returns ApplicationInsightsApplication.

```
Update-ApplicationInsightsMonitoring -Name appName [-InstrumentationKey "0000000-0000-000-
000-0000"
```

]

- `-Name`: The name of a web app in IIS.
- `-InstrumentationKey` (Optional.) Use this to change the resource to which the app's telemetry is sent.
- This cmdlet:
  - Upgrades the named app to the version of the SDK most recently downloaded to this machine. (Only works if `SdkState==EnabledAfterDeployment`)
  - If you provide an instrumentation key, the named app is reconfigured to send telemetry to the resource with that key. (Works if `SdkState != Disabled`)

`Update-ApplicationInsightsVersion`

- Downloads the latest Application Insights SDK to the server.

Questions about Status Monitor

What is Status Monitor?

A desktop application that you install in your IIS web server. It helps you instrument and configure web apps.

When do I use Status Monitor?

- To instrument any web app that is running on your IIS server - even if it is already running.
- To enable additional telemetry for web apps that have been built with the Application Insights SDK at compile time.

Can I close it after it runs?

Yes. After it has instrumented the websites you select, you can close it.

It doesn't collect telemetry by itself. It just configures the web apps and sets some permissions.

What does Status Monitor do?

When you select a web app for Status Monitor to instrument:

- Downloads and places the Application Insights assemblies and .config file in the web app's binaries folder.
- Modifies `web.config` to add the Application Insights HTTP tracking module.
- Enables CLR profiling to collect dependency calls.

Do I need to run Status Monitor whenever I update the app?

Not if you redeploy incrementally.

If you select the 'delete existing files' option in the publish process, you would need to re-run Status Monitor to configure Application Insights.

What telemetry is collected?

For applications that you instrument only at run-time by using Status Monitor:

- HTTP requests
- Calls to dependencies

- Exceptions
- Performance counters

For applications already instrumented at compile time:

- Process counters.
- Dependency calls (.NET 4.5); return values in dependency calls (.NET 4.6).
- Exception stack trace values.

Learn more

Video

09:40

Next steps

View your telemetry:

- Explore metrics to monitor performance and usage
- Search events and logs to diagnose problems
- Analytics for more advanced queries
- Create dashboards

Add more telemetry:

- Create web tests to make sure your site stays live.
- Add web client telemetry to see exceptions from web page code and to let you insert trace calls.