Activate Microsoft Azure with Azure Application Insights

Module Overview

Application Insights - Advanced

Agenda

Module
Application Insights - Advanced

Section 1: Fundamental Concept

Lesson 1: Application Insights Core

Session

Definition

- Start when user starts the app & finished when user leaves

Goal

- Strive to associate every telemetry event with a specific user session
- Provide a good measure of popularity contexts : Device, OS, …

Internals

- If instrument client & server, SDK will propagate session id between client & server
- Enable easy correlation between pageview and request

Defaults

- Renewal : terminates after 30 mins of inactivity
- Expiration: terminates 24 h of activity

User

Definition

- By default, user is identified by placing a cookie
- Each session is associated with a unique user id
- User who uses multiple browsers or devices will be counted more than once

New users

- Counts the users whose first sessions with the app occurred during this interval.

Authenticated users

- You can get a more accurate count by providing Application Insights with a unique user identifier
    - appInsights.setAuthenticatedUserContext(userId);

Dependencies

- What is a dependency ?
  - Definition
  - Dependency monitor

ASP.NET

- SQL databases
- ASP.NET web and WCF services that use HTTP-based bindings
- Local or remote HTTP calls
- Azure DocumentDb, table, blob storage, and queue

Java

- Calls to a database through a JDBC driver, such as MySQL, SQL Server, PostgreSQL or SQLite.

Web pages

- AJAX calls

Execution mode

Different execution mode when running Application Insights in :

Debug : Development Mode

- Instantaneous send

Release : Production Mode

- Batch send of data

AI Agent – Outgoing Servers Firewall

Telemetry - needed all the time

- dc.services.visualstudio.com:80
- dc.services.visualstudio.com:443
- dc.applicationinsights.microsoft.com

Configuration - needed only when making changes

- management.core.windows.net:443
- management.azure.com:443
- login.windows.net:443
- login.microsoftonline.com:443
- secure.aadcdn.microsoftonline-p.com:443
- auth.gfx.ms:443
- login.live.com:443

Installation

- packages.nuget.org:443

Availablity Test

Need to whitelist a list of Ips

- Ports : 443 / 80
- Details : https://docs.microsoft.com/en-us/azure/azure-monitor/app/ip-addresses#outgoing-ports

Need External Access

- Your Application need to be reachable
  - On prem (May require action)
  - Cloud (Out of the box)

Module
Application Insights - Advanced

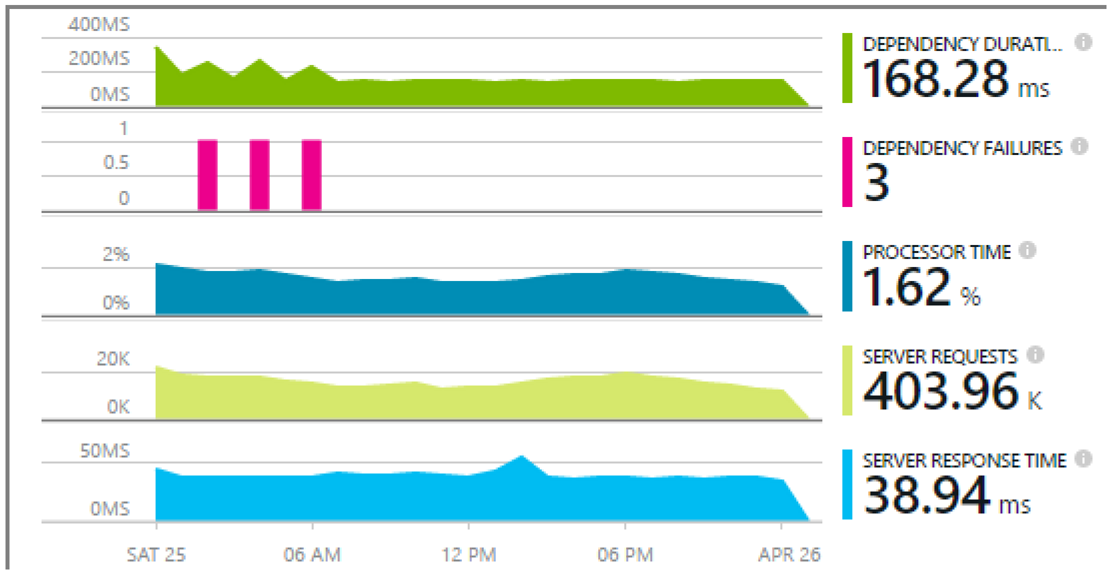
Section 2: Application Insight Agent (Status Monitor)

Lesson 1: Status Monitor Essentials

Application Insights Agent (Status Monitor)

Enable you diagnose exceptions and performance issues in web applications :

- Instrument asp.net apps
- Send data to Application Insights



Install on any IIS Server :

- Manually (using WPI)
- Using Extension (Azure VM, Azure Web Apps)
- Powershell: Install-Module -Name Az.ApplicationMonitor -RequiredVersion 1.0.1

Note: *In the latest version of .Net most of the data collected by AISM are directly collected by SDK*

How does it work

Status Monitor Internals

- AISM add Microsoft.AI.HttpModule.dll assembly in the application bin folder
- Dll is loaded automatically by IIS pipeline
- And the same HTTP module is registered in the run-time.

Note

- This behavior may change in future version, hence, this is only for information and for the current version
- E.G. : Old version of AISM used to insert HTTP Module in web.config

Configuration

Server OS

- Windows Server 2008
- Windows Server 2008 R2
- Windows Server 2012
- Windows Server 2012 R2
- Windows server 2016
- Windows server 2019

Client OS

- Windows 7
- Windows 8
- Windows 8.1
- Windows 10

.NET Framework

- 4.5 +

IIS

- Supported version IIS 7, 7.5, 8, 8.5

Example of Common Scenario

Line of Business applications

- Source code not available anymore
- No access to Source code

3rd Party Solutions

- Microsoft Dynamic CRM
- Microsoft SharePoint

Module
Application Insights - Advanced

Section 2: Application Insight Status Monitor

Lesson 2: Troubleshooting

Application Insights Status Monitor (AISM)

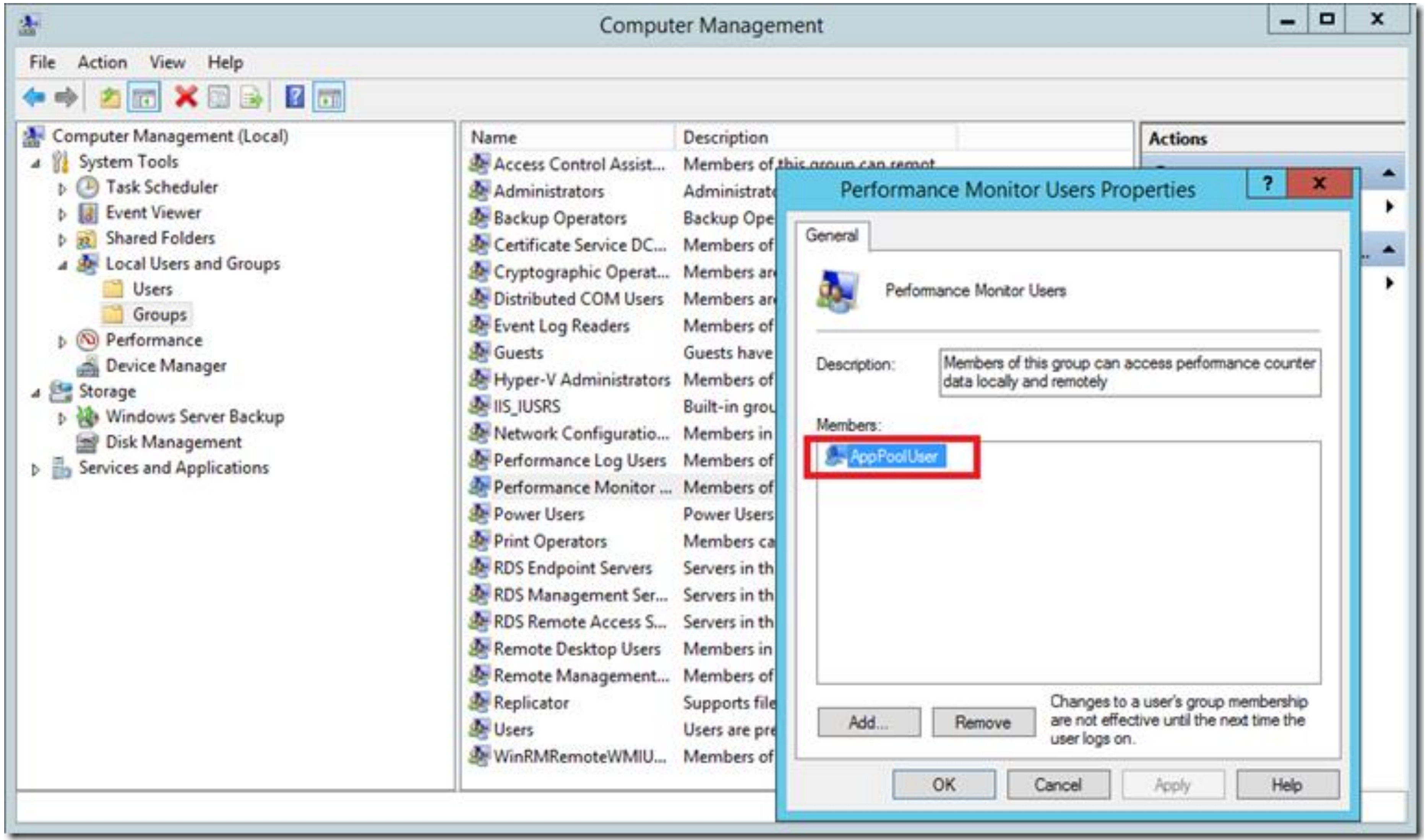
No telemetry data

- Can have several root cause:
  - Network
  - IIS
  - Rights (Authorizations, etc)

Ensure

- Application Pool Identity is a member of « **Performance Monitor Users** » group
- Update to the latest version of Status Monitor

The application 'Default Web Site/FabrikamProd' running under application pool 'DefaultAppPool' (security principal 'IIS AppPool\DefaultAppPool') might have insufficient permissions to collect performance data. If no performance counter data is visible in the portal, make sure that the security principal under which the application runs is a member of 'Performance Monitor Users' group.



Name	Description
Access Control Assist...	Members of this group can remot
Administrators	Administrat
Backup Operators	Backup Ope
Certificate Service DC...	Members of
Cryptographic Operat...	Members ar
Distributed COM Users	Members ar
Event Log Readers	Members of
Guests	Guests have
Hyper-V Administrators	Members of
IIS_IUSRS	Built-in grou
Network Configuratio...	Members in
Performance Log Users	Members of
Performance Monitor ...	Members of
Power Users	Power Users
Print Operators	Members ca
RDS Endpoint Servers	Servers in th
RDS Management Ser...	Servers in th
RDS Remote Access S...	Servers in th
Remote Desktop Users	Members in
Remote Management...	Members of
Replicator	Supports file
Users	Users are pre
WinRMRemoteWMIU...	Members of

Performance Monitor Users Properties

General

Performance Monitor Users

Description: Members of this group can access performance counter data locally and remotely

Members:

AppPoolUser

Add... Remove

Changes to a user's group membership are not effective until the next time the user logs on.

OK Cancel Apply Help

Troubleshooting Traffic

Note

- Nothing is traced in the event log

Alternative

- Traffic being sent can be checked in Fiddler
- Look for the following destination : ***“dc.services.visualstudio.com/v2/track”***

Troubleshooting Traffic (continued)

Goal

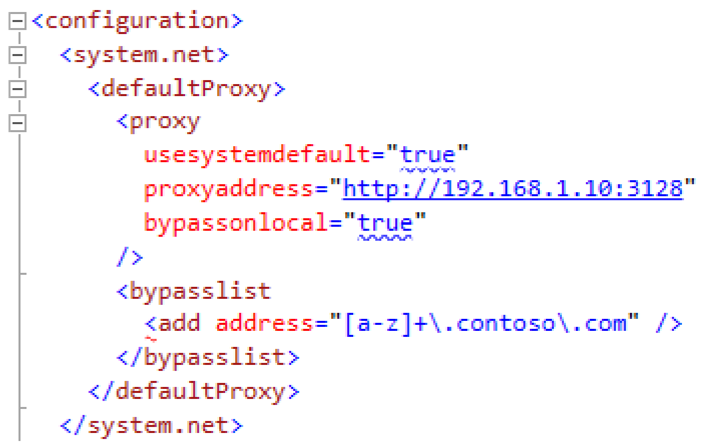
- Data collected using SDK can be checked by looking at ETW events at ETW events collected using PerfView

Command

- **PerfView.exe** /onlyProviders=*Microsoft-ApplicationInsights-Extensibility-Web,*Microsoft-ApplicationInsights-Web,*Microsoft-ApplicationInsights-Core,*Microsoft-ApplicationInsights-Extensibility-DependencyCollector,*Microsoft-ApplicationInsights-Extensibility-Rtia-SharedCore,*Microsoft-ApplicationInsights-Extensibility-WindowsServer,*Microsoft-ApplicationInsights-WindowsServer-TelemetryChannel collect

Realworld Gotcha

- Proxy
  - Common use case in enterprise context
  - Can be one of the first root cause for « **no telemetry** »
- Solution
  - In Asp.net Web.config
  - Configures the Hypertext Transfer Protocol (HTTP) proxy server



```
<configuration>
  <system.net>
    <defaultProxy>
      <proxy
        usesystemdefault="true"
        proxyaddress="http://192.168.1.10:3128"
        bypassonlocal="true"
      />
      <bypasslist
        <add address="[a-z]+\.contoso\.com" />
      </bypasslist>
    </defaultProxy>
  </system.net>
```

Module
Application Insights - Advanced

Section 3: Application Insights SDK

Lesson 1: SDK Essentials

API Method

TrackPageView	• Pages, screens, blades or forms
TrackEvent	• User actions and other events. Used to track user behavior or to monitor performance
TrackMetric	• Performance measurements such as queue lengths not related to specific events
TrackException	• Log exceptions for diagnosis. Trace where they occur in relation to other events and examine stack traces
TrackRequest	• Log the frequency and duration of server requests for performance analysis
TrackTrace	• Diagnostic log messages. You can also capture 3rd-party logs
TrackDependency	• Log the duration and frequency of calls to external components on which your app depends

Supported Platform

Popular

- ASP.NET Web Server SDK
- .NET Core SDK
- .NET Logging Adapters
- ASP.NET 5
- Android
- iOS
- Java
- JavaScript
- Windows Phone and Store C#|VB
- Windows Phone and Store C++
- Visual Studio tools

Other platforms

- C++ Universal apps
- Go
- iOS
- Node.js
- OSX
- PHP
- Python
- Ruby
- StatsD
- WordPress
- Xamarin

How to

TrackTrace

- custom trace for dynamic/punctual to fine-tuned query, investigations
- Ex: any logs, warning, etc.

TrackMetric

- Track anything that is purely numeric
- ex : e-commerce selling t-shirt would be tracking number of sold items…

TrackRequest

- Collect information about a request outside of the usual application workflow.

Measurements and Properties

Measurements

- **Numeric values that can be presented graphically**
- Value should be >= 0
- Feature:
  - Display in graph
  - Segmented by properties
  - Aggrate (Sum, Average, etc)

Properties

- **String values that you can use to filter your telemetry in the usage reports**
- 1K data limit
- Feature:
  - Grouping
  - Filtering

Page views

Typical usage

- All type of solution with Front end
- Single Page Application

Goal

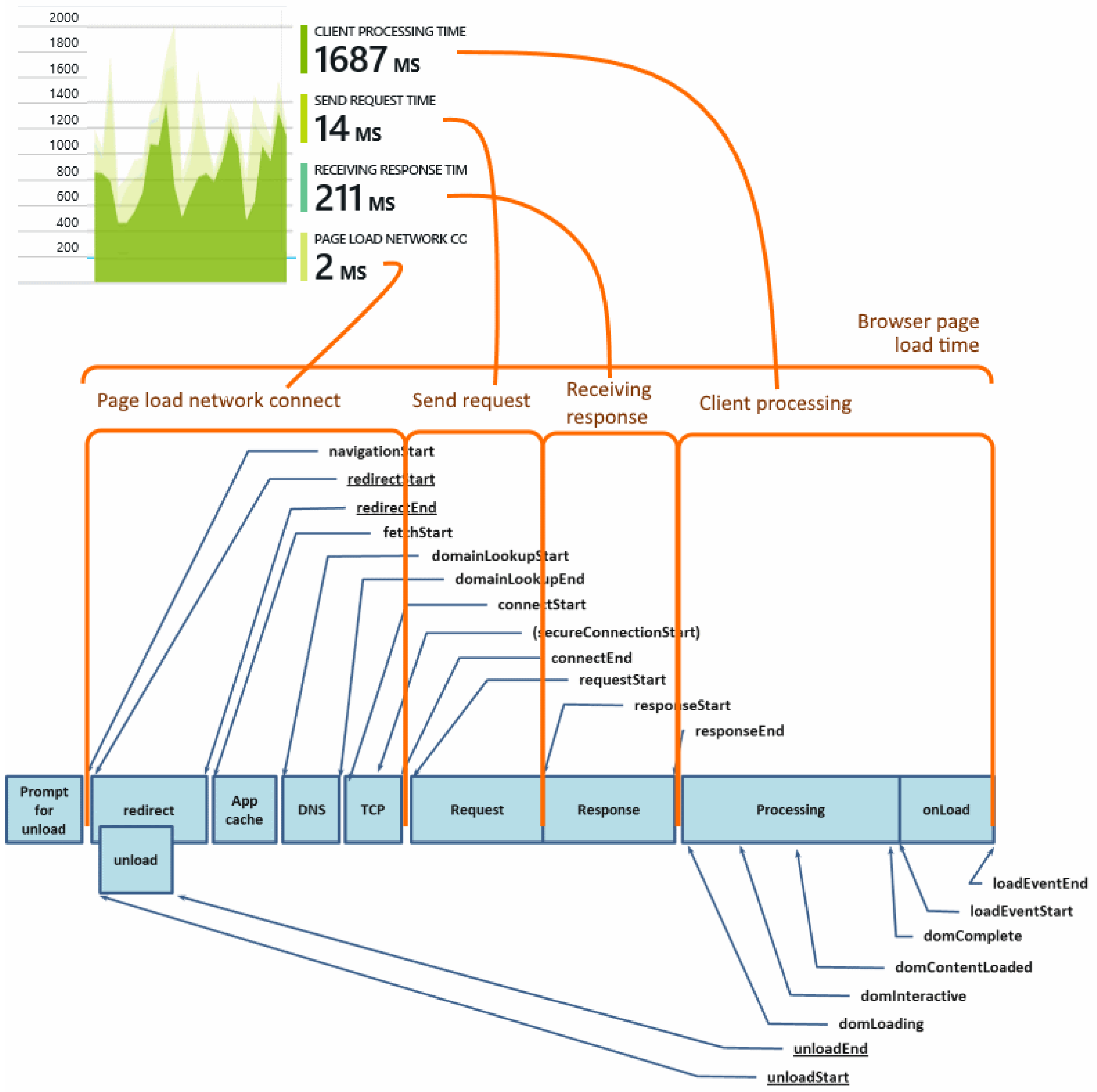
- Use JavaScript to record each “new page” browsing
- Collect Client Traces

Examples

- Online website with mini games
- instead of having each game in a separate webpage, all of them are loaded with JavaScript (quick switch to maximize user experience)
- Use telemetry module in JavaScript to track that usage

```
telemetryClient.trackPageView(game.Name);
```

Page load performance



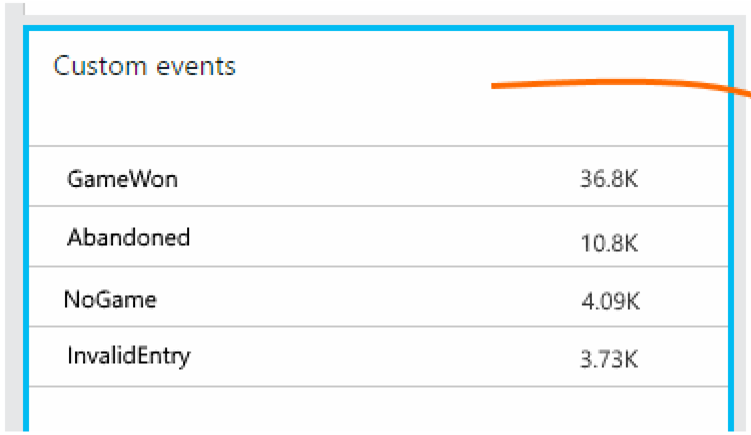
Custom Events

Need to understand how your application is being used

Can define custom event to track:

- features
- scenarios
- any actions you wish to monitor

```
var tc = new Microsoft.ApplicationInsights.TelemetryClient();
tc.TrackEvent("GameEnd");
```



Custom events

GameWon	36.8K
Abandoned	10.8K
NoGame	4.09K
InvalidEntry	3.73K

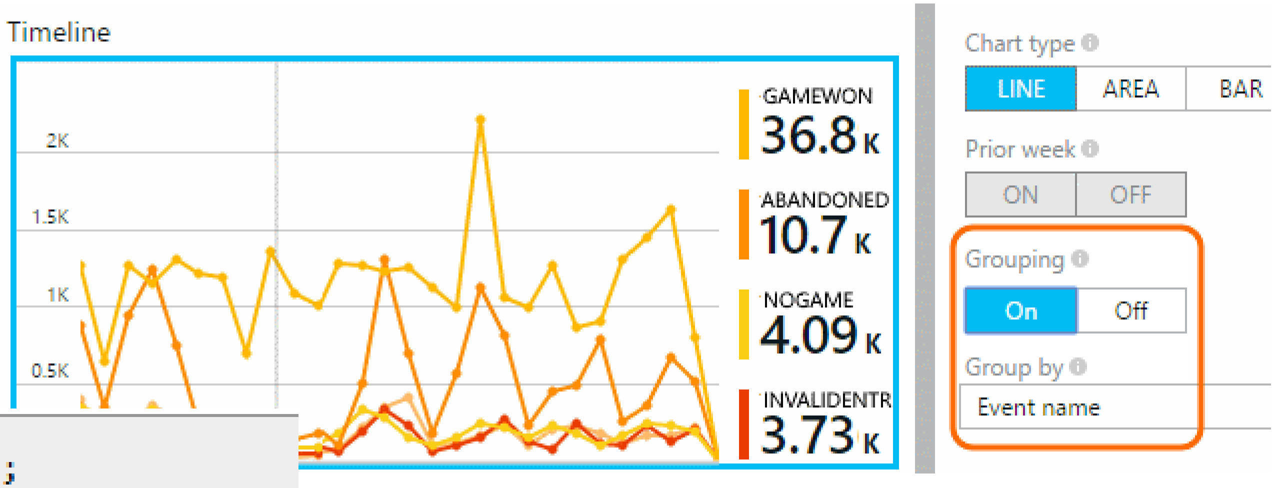


Chart type: [✓ LINE] AREA BAR

Prior week: ON OFF

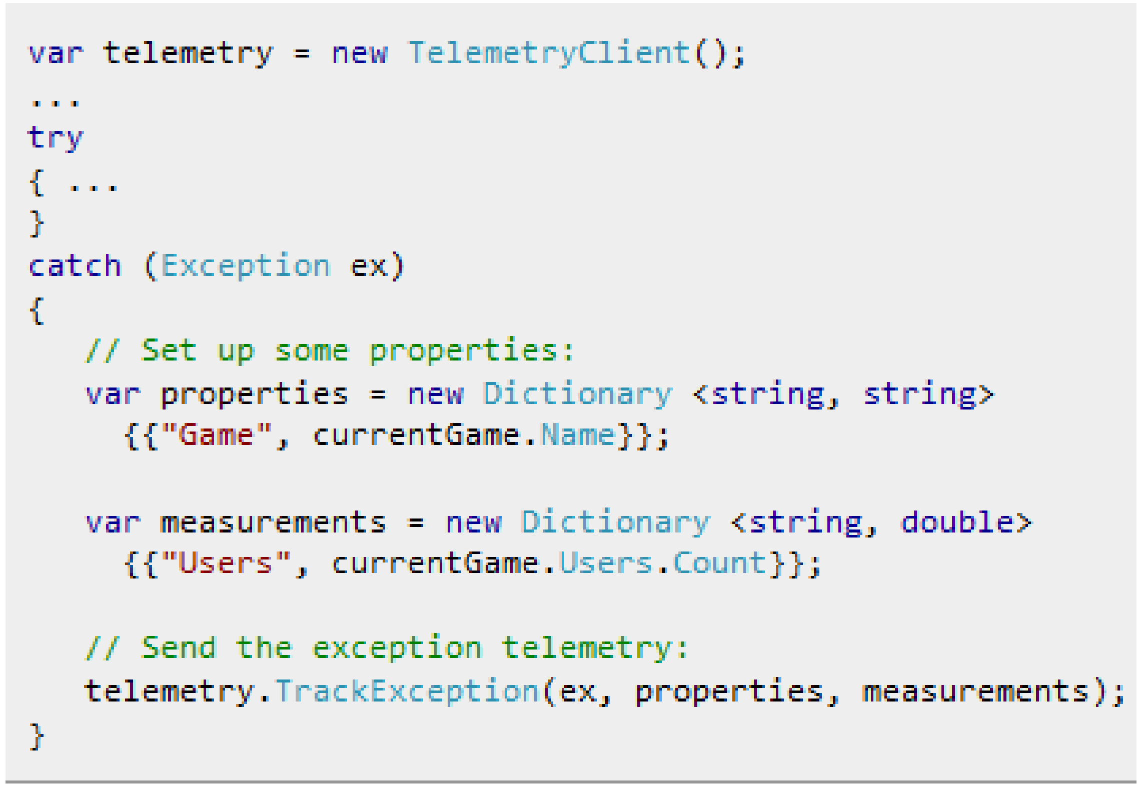
Grouping: [✓ On] Off

Group by: Event name

```
var successCode = AttemptTransfer(transferAmount, ...);
if (successCode < 0)
{
   var properties = new Dictionary <string, string>
        {{ "Code", returnCode, ... }};
   var measurements = new Dictionary <string, double>
     {{"Value", transferAmount}};
   telemetry.TrackEvent("transfer failed", properties, measurements);
}
```

Exception

- Capture full stack trace
- Collect additional data using properties & measurements



```
var telemetry = new TelemetryClient();
...
try
{ ...
}
catch (Exception ex)
{
   // Set up some properties:
   var properties = new Dictionary <string, string>
     {{"Game", currentGame.Name}};

   var measurements = new Dictionary <string, double>
     {{"Users", currentGame.Users.Count}};

   // Send the exception telemetry:
   telemetry.TrackException(ex, properties, measurements);
}
```

Diagnostics Trace

Support

- log4J
- Log4net
- Nlog
- System.Diagnostics.Trace

Enable easy correlation

- Logs traces
- User actions
- Exceptions
- Other events

Using Telemetry Initializer

- Create a custom Initializer :
  - Using ITelemetryInitializer

```
public class MyTelemetryInitializer : ITelemetryInitializer
{
  public void Initialize(ITelemetry telemetry)
  {
      var requestTelemetry = telemetry as RequestTelemetry;
      // Is this a TrackRequest() ?
      if (requestTelemetry == null) return;
      int code;
      bool parsed = Int32.TryParse(requestTelemetry.ResponseCode, out code);
      if (!parsed) return;
      if (code >= 400 && code < 500)
      {
          // If we set the Success property, the SDK won't change it:
          requestTelemetry.Success = true;
          // Allow us to filter these requests in the portal:
          requestTelemetry.Context.Properties["Overridden400s"] = "true";
      }
      // else leave the SDK to set the Success property
  }
}
```

```
// Telemetry initializer class
public class MyTelemetryInitializer : ITelemetryInitializer
{
    public void Initialize (ITelemetry telemetry)
    {
        telemetry.Properties["AppVersion"] = "v2.1";
    }
}
```

DEMO:
Using Application Insights
ASD

Module
Application Insights - Advanced

Section 4: Application Insights Configuration

Lesson 1: Overview

Overview

Goal

- Provide control over data collection process
- Can enable or disable telemetry modules and initializers, and set parameters for some of them.

File name

- ApplicationInsights.config or ApplicationInsights.xml
- Depending on the type of your application.

Automatically added

- When you install most versions of the SDK
- Also added to a web app by Status Monitor on an IIS server
- or when you select the Application Insights extension for an Azure website or VM.

Components

Telemetry Modules (ASP.NET)	• Each module collects a specific type of data and uses the core API to send the data. • Modules are installed by different NuGet packages, which also add the required lines to the .config file.
Telemetry Channel	• Manages buffering and transmission of telemetry to the Application Insights service.
Telemetry Initializers (ASP.NET)	• Set context properties that are sent along with every item of telemetry.
Telemetry Processors (ASP.NET)	• Can filter and modify each telemetry item just before it is sent from the SDK to the portal.

Adding performance counters

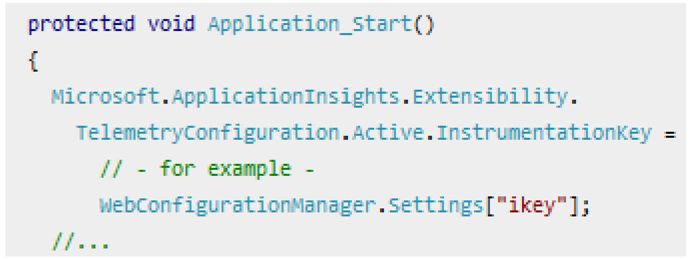
- Edit ApplicationInsights.config

```
<ApplicationInsights xmlns="http://schemas.microsoft.com/ApplicationInsights/2013/Settings">
  <TelemetryModules>
    <Add Type="Microsoft.ApplicationInsights.Extensibility.DependencyCollector.DependencyTrackingTelemetryModule,
Microsoft.ApplicationInsights.Extensibility.DependencyCollector" />
    <Add Type="Microsoft.ApplicationInsights.Extensibility.PerfCounterCollector.PerformanceCollectorModule,
Microsoft.ApplicationInsights.Extensibility.PerfCounterCollector">
      <Counters>
        <Add PerformanceCounter="\ASP.NET\Requests Rejected"/>
        <Add PerformanceCounter="\ASP.NET\Application Restarts"/>
      </Counters>
    </Add>
    <Add Type="Microsoft.ApplicationInsights.Extensibility.Implementation.Tracing.DiagnosticsTelemetryModule,
Microsoft.ApplicationInsights" />
    <Add Type="Microsoft.ApplicationInsights.Extensibility.Web.RequestTrackingTelemetryModule,
Microsoft.ApplicationInsights.Extensibility.Web" />
    <Add Type="Microsoft.ApplicationInsights.Extensibility.Web.ExceptionTrackingTelemetryModule,
Microsoft.ApplicationInsights.Extensibility.Web" />
    <Add Type="Microsoft.ApplicationInsights.Extensibility.Web.DeveloperModeWithDebuggerAttachedTelemetryModule,
Microsoft.ApplicationInsights.Extensibility.Web" />
  </TelemetryModules>
```

InstrumentationKey

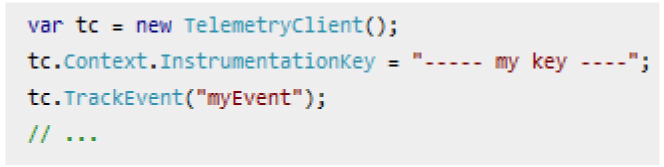
- Determines the Application Insights resource in which your data appears.
- create a separate resource, with a separate key, for each of your applications.
- Can be Set dynamically

- Ex for Asp.net (Global.aspx.cs)



```
protected void Application_Start()
{
  Microsoft.ApplicationInsights.Extensibility.
    TelemetryConfiguration.Active.InstrumentationKey =
      // - for example -
      WebConfigurationManager.Settings["ikey"];
  //...
```

- Can setup for a specific set of event – for a specific TelmetryClient :



```
var tc = new TelemetryClient();
tc.Context.InstrumentationKey = "----- my key ----";
tc.TrackEvent("myEvent");
// ...
```

Module
Application Insights - Advanced

Section 5: Reducing data traffic

Lesson 1: Tips

Tips for reducing your data rate

Tuning

- Use Sampling

Ajax

- Limit the number of Ajax calls that can be reported in every page view, or switch off Ajax reporting.

SDK Modules

- Switch off collection modules you don't need by editing ApplicationInsights.config.

Pre-aggregate metrics.

- TrackMetric – overload with calculation of average
- Or you can use a pre-aggregating package.

Sampling

Definition

- Method of reducing the rate at which telemetry is sent to your app
  - Can help to avoid throttling
  - Available in different “flavor” : adaptative, fixe, ingestion

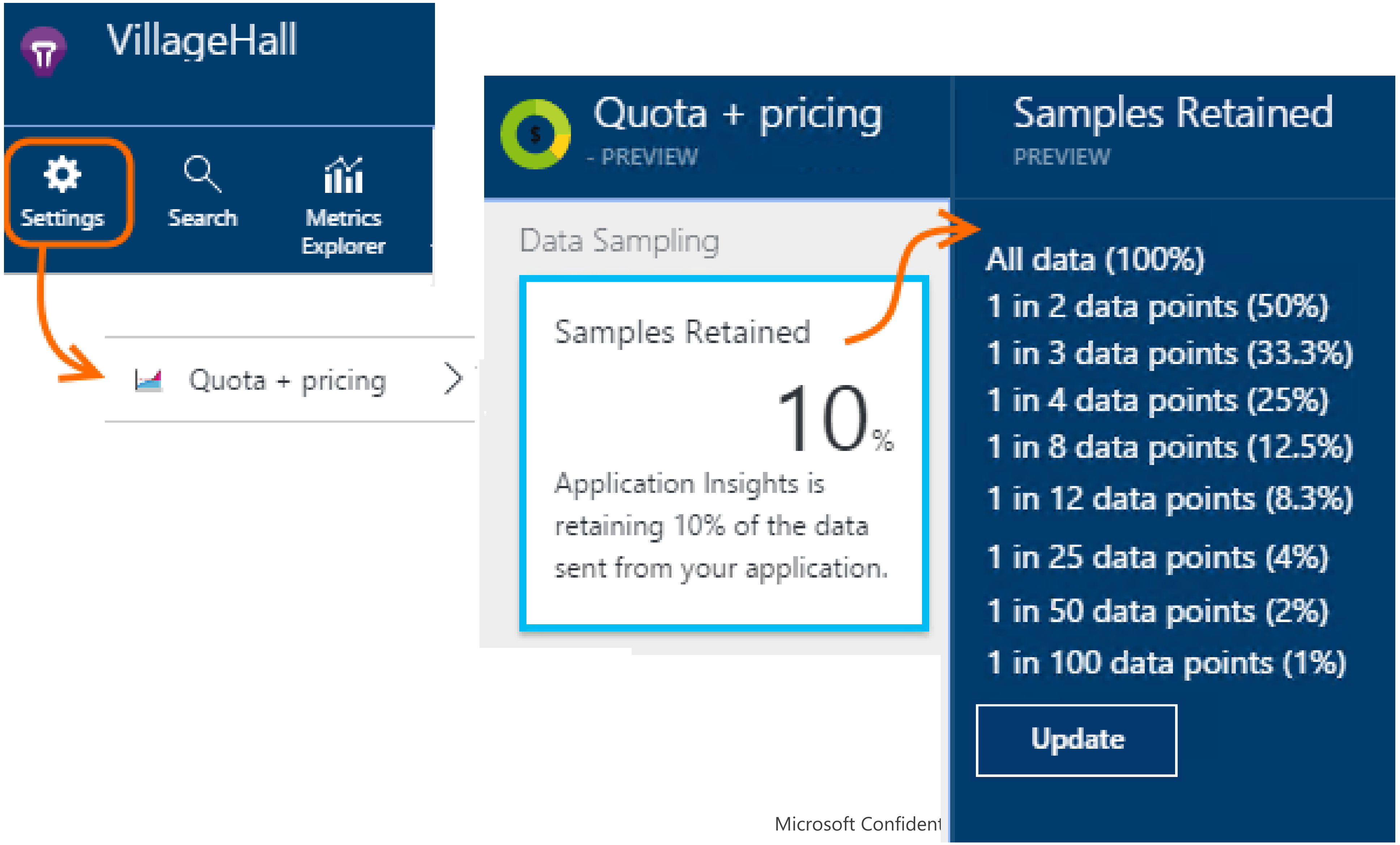
Recommendation

- Adaptive sampling, which automatically adjusts to the volume of telemetry that your app sends.
- Operates in the SDK in your web app, so that the telemetry traffic on the network is reduced.

Alternative

- Can set *ingestion sampling* on the Quotas + pricing blade.
- Operates at the point where telemetry from your app enters the Application Insights service.
  - Doesn't affect the volume of telemetry sent from your app
  - Reduces the volume retained by the service.

Ingestion Sampling



Application Insights is retaining 10% of the data sent from your application.

All data (100%)
1 in 2 data points (50%)
1 in 3 data points (33.3%)
1 in 4 data points (25%)
1 in 8 data points (12.5%)
1 in 12 data points (8.3%)
1 in 25 data points (4%)
1 in 50 data points (2%)
1 in 100 data points (1%)

Module
Application Insights - Advanced

Section 6: Best Practices

Lesson: Best Practices

SDK Best Practices

Reuse

- Do not to construct a new TelemetryClient every time
- Recommendation is to reuse the client instance

Dedicated

- Create separate instances when you need to initialize with different configuration

Leverage Defaults

- Default TelemetryClient’s constructor without parameter will use AppInsights.config file
- TelemetryClient.Active

General guideline

Application Insights resource type

- Recommendation is to not combine Web & API Apps or any other specific dashboard
- Otherwise will end up seeing things that doesn't make sense or missing data

Two step process

- Start first Proof of concept (POC) / or first release
- Optimize Data sent / Refactoring

Microsoft