10

Monitoring and optimizing

the application

Application monitoring is a simple process of keeping track of various aspects of an application and how they are performing. It is very important for consistent quality check and improvement and important to find out problems in the application before it gets to users.  
Application monitoring will not only let us know the performance of the application and issues within the application but also keep in check the status of its related databases and APIs.

In this chapter, we’ll be discussing about various ways of application monitoring. High level topics that we’ll be covering in this chapter:

* API Level Monitoring and Various Tools for API Monitoring
* Monitoring Application with Test Cloud
* Monitoring Application using Android Monitoring Tools

**API Level Monitoring**APIs (Application Programming Interface), are an integral part of today’s integrated development environments. It can also be understood with client-server relationship. Where Mobile app is the client requesting for a resource and API is at the server side that has a URL for any application that wants to make a request.  
Most of the applications share common APIs between web and mobile. APIs are a great way of providing a consistent operation behavior across different platforms. It also helps share the same business and data layer operations between different mobile application platforms like you can use the same APIs for iOS, Android and web applications.

Because API is so important in the development of a mobile application, it becomes equally critical to monitor APIs to ensure high availability. API goes down, entire application can stop working and user might not be able to do any operation that requires API to be available, that usually is any server operation and not offline.

**Why API Monitoring is Critical**

When we talk about continuous integration, it should include a continuous cycle of development, testing, deployment and monitoring. As engineering teams like using Agile practices and other techniques for software development, testing and deployment, it is critical that monitoring becomes a must have part of that continuous integration cycle and we update our monitoring methods and tools with the correct and latest versions to ensure high availability for our users and other consumers like mobile and web applications that might be using our APIs.

As we discussed earlier in this chapter, when APIs fail, your applications fail. If you are using APIs to provide services for your mobile app and they are either not available, functioning improperly or not being responsive, it impacts the performance of your app directly and the end user experience gets compromised.

Also, it is very important to monitor APIs not only that you are developing but also, third party ones that you might be using in the application.

**Important factors in API monitoring**

When monitoring API there are some key points or areas that need to be covered to ensure better availability.

* **API Availability**  
  We need to make sure if the API is available or not, sometimes the server might be down for some reason or the connection can be interrupted based on location and server.
* **Quality of Response**  
  When called an API, how is the quality of response returned from the API. Is it according to the agreement or not.
* **Response Time**   
  What is the response time to get a result when called the API.

**Developer’s Role in handling API unavailability**

It is also a good idea for an application developer to keep in mind that APIs might not be available some time and write code in a way that handles these kinds of situations gracefully. Even when application is not able to perform some API operations, if it is informed to the user in a nice way, it can help the user experience much more than facing runtime exceptions. Writing code to handle API exceptions and to handle scenarios where API does not respond.

**Various Tools for API Monitoring**

There are many tools available in the market for API monitoring and testing. What to choose totally depends on what you want to achieve through those tools. Some tools provide great support for performance monitoring and other tools are better suited for quality testing and recognizing erroneous data.

**Postman**

Postman is a rest client that started off as a Chrome browser plugin but recently came out with native versions for both Mac and Windows.

At a high level, you can use it to send a post request to your web server and it gives you the response back. It allows you to set up all the headers and cookies your API expects, and then check the response when it comes back.

* Can be used for both automated and exploratory testing
* Can be run on Mac, Windows, Linux &Chrome Apps
* Has a bunch of integrations like support for Swagger & RAML formats
* Has Run, Test, Document and Monitoring Features
* Doesn’t require learning a new language

**Karate DSL**

Karate allows you to create a test that can sequence calls to any kind of web-service and assert that the responses are as expected.

* Build on top of Cucumber-JVM
* Can run test and generate reports like any standard Java project
* Test can be written without any Java knowledge required
* Tests are easy to write even for non-programmers
* Check out a quick example on how to get started using Karate with BDD.

**SoapUI**

SoapUI is a headless functional testing tool from SmartBear software. It comes in two flavors: Free open source version and Pro Version. Since the free version is open-source, you can actually gain access to the full source code and modify as needed. The pro version is user-friendlier, and has additional functionality including a form editor, an assertion wizard for Xpath, and SQL query builder. The free version lets you:

* Can easily create custom code using Groovy
* Drag and Drop Test Creating
* Can create complex scenarios
* Asynchronous Testing
* SoapUI’s Mock Service lets you mimic web services before they are implemented

**HttpMaster Express**

HttpMaster describes itself as a web development and test tool to automate testing of web sites and services. It can be used to test RESTful web services and API applications. HttpMaster also allows you to and monitor API responses.

* HttpsMaster project offers global options to customize your API request
* Parameter capabilities enable you to include dynamic data with your request
* You can use request chaining to leverage request items to include some data from previous request with the next request

**Rest- Assured**

Rest-Assured is an open-source Java Domain-specific language (DSL) that makes testing REST service simple. It simplifies things by eliminating the need to use boiler-plate code to test and validate complex responses. It also supports XML and JSON Request/Responses.

* Removes need to create boilerplate code required to interact with a rest service
* Support BDD Given/When/Then syntax
* Integrated seamlessly with Java projects

**RestSharp**

RestSharp is a simple REST and HTTP API Client for .NET

* Supports .NET 3.5+, Silverlight 5, Windows Phone 8, Mono, MonoTouch, Mono for Android
* Easy installation using NuGet for most .NET flavors
* GET, POST, PUT, PATCH, HEAD, OPTIONS, DELETE supported

**Rest Console Console**

HTTP Client and Request Visualizer and Constructor tool, helps developers build, debug and test RESTful APIs. Rest Console is a HTTP Request Visualizer and Constructor tool, helps developers build, debug and test RESTful APIs.

* Easy query parameters creation
* Syntax highlighting
* Authentication support: Plain, Basic, OAuth + Custom

**Hippie-Swagger**

hippie-swagger is a tool for testing RESTful APIs. It’s also an API testing tool with automatic swagger assertions. In addition to validating API behavior, it will fail tests when swagger documentation is missing or inaccurate.

* Can validate All aspects of swagger file validated; parameters, request/response body, paths, etc.
* Accurate, human readable assertion messages

**Airborne**

Airborne is an open source Ruby based RSpec driven API testing framework.

* Works with Rack application like Sinatra and Grape
* Works with APIs written in Rails

**Mockbin**

Mockbin was recommended by Augusto Marietti. Mockbin allows you to generate custom endpoints to test, mock, and track HTTP requests & responses between libraries, sockets and APIs.

* Mock Custom Endpoints
* Create Custom HTTP Methods
* Log and inspect incoming calls to your custom endpoints

**Using Test Cloud for Monitoring**

You learned about Xamarin Test Cloud in previous chapters and how to use it for continuous testing in the continuous integration life cycle. Here we will discuss more about how to use Xamarin Test Cloud and the analytics it provides after running application on different set of devices.

We will be using 3 applications here to see the monitoring analytics and compare them to get a better understanding of how this helps us identify various performance and functionality related issues in our application.

Applications we will be using for the walkthrough:

1. PhoneCallApp (The application we developed in the previous chapters)
2. Xamarin Store (Sample Android application provided by Test Cloud)
3. Xamarin Store for iOS (Sample iOS application provided by Test Cloud)

Xamarin Test Cloud can help us identify application functionality related issues on real devices.  
It is a great source of application monitoring in terms of testing on different mobile devices and with different versions of operating systems.

Getting a detailed analysis of various applications functions is very important to make sure our application is running as expected on our target devices.  
With that being said, it is also critical to the application to be able to run on different operating system versions and on to analyze how it performs and how much memory usage does it have.

**Benefits of Monitoring with Test Cloud**

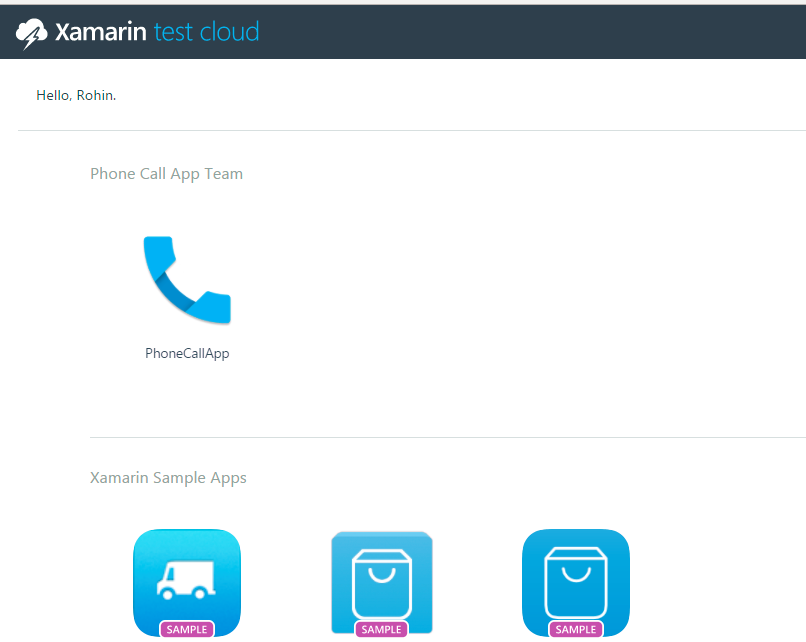
With Test cloud it not only provides monitoring capabilities but also relieves us from testing the same application functionality on different devices manually and thus giving a true continuous integration process.

* Provides continuous testing capabilities to our CI process with automated test runs and detailed reports with notifications.
* Testing application on different OS versions is critical to the success of a mobile application and Test Cloud serves that purpose very well.
* Testing application on different devices from its huge device list available on cloud
* Analyzing performance of the application
* Analyzing memory usage on different devices with different hardware configurations

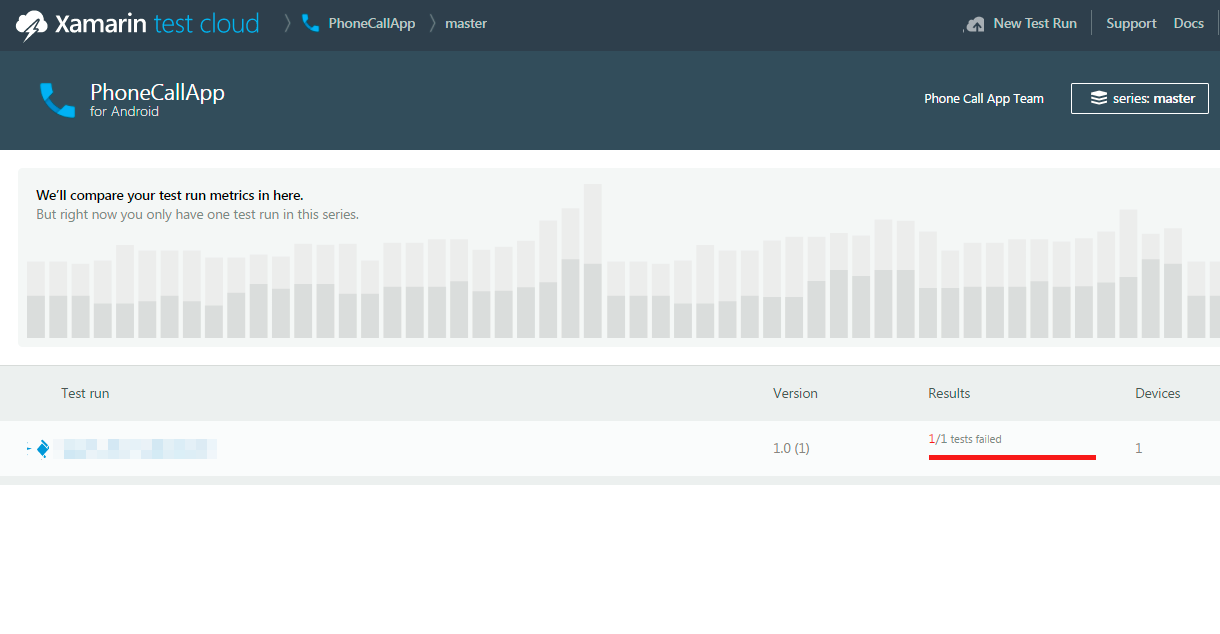
**PhoneCallApp (Android)**

Let’s go through some steps to see how to monitor our PhoneCallApp.

1. Go to <https://testcloud.xamarin.com/>
2. Click on the PhoneCallApp icon to get to the details of “Test Runs”.



1. On the next page you’ll see a list of tests run for the application.



1. Now because we have only one test run as of now, Test Cloud does not provide us with the graphical metrices above. In other examples we see next, you’ll be able to see more detailed comparison of different test runs.
2. Click on the test run from the list to see it’s results.