عنوان: **معرفی سرویس‌های ارائه شده توسط شرکت‌های گوگل، آمازون و مایکروسافت در قالب رایانش ابری - قسمت اول**
نویسنده: مهدی نقدی
تاریخ: ۱۳۹۲/۰۹/۱۲ ۹:۲۵
آدرس: www.dotnettips.info
برچسب‌ها: Cloud computing, google analytics, GAE

رایانش ابری مفهوم نسبتا جدیدی در عرصه‌ی فناوری اطلاعات است و در حال گسترش می-باشد. به طور خلاصه رایانش ابری به همه چیز اعم از برنامه کاربردی(Application)، سکو ی (Platform) توسعه نرم افزار، سخت افزار و زیرساخت، به عنوان سرویس نگاه می‌کند. زیرساخت های موجود در مراکز داده(Data Center) به انضمام نرم‌افزارهایی که در آن قرار دارند، مجموعه-ایی را تشکیل می-دهند که ابر نامیده می-شود. به عبارت ساده-تر رایانش ابری یعنی استفاده- اشتراکی از برنامه-ها و منابع یک محیط شبکه-ای برای انجام یک کار، بدون این که مالکیت، مدیریت منابع شبکه و سخت- افزار و برنامه-ها، برای استفاده کننده مهم باشد. در رایانش ابری منابع کامپیوترها، برای انجام یک کار استفاده می‌شوند و داده-های مربوط به پردازش، در هیچ کدام از کامپیوترهای شخصی ذخیره نمی-شوند، بلکه در جای دیگری در داخل همان منابع شبکه، ذخیره می-شوند تا در زمان و مکان دیگری قابل دسترسی باشند.

بر همین اساس شرکت های پیشرو در زمینه فناوری اطلاعات به ارائه سرویس هایی تحت عنوان خدمات رایانش ابری پرداخته اند و هدف از این سری مطالب ارائه شده، شرح مختصری بر سرویس‌های ارائه شده می باشد. در قسمت اول به معرفی سرویس های شرکت گوگل پرداخته می شود و در قسمت‌های بعدی، سرویس‌های شرکت‌های مایکروسافت و آمازون معرفی می‌گردد.

سرویس-های رایانش ابری گوگل، در زیر دو چتر قرار دارند. گروه اول شامل مجموعه گسترده-ای از برنامه-های محبوب گوگل مانند Google Earth ، Google Mail ، Google Health ، Google Doc هستند که با کلیک بر روی گزینه More و Even More که در بالای صفحه اصلی گوگل قرار دارند، می-توان به آن-ها دسترسی پیدا کرد.

دومین محصول مبتنی بر ابر گوگل، ابزار توسعه PaaS گوگل است. این سکو در سال 2008 برای توسعه برنامه-های تحت وب، با استفاده از زیرساخت گوگل به نام موتور Google App معرفی شد. هدف از آن قادر ساختن توسعه دهندگان برای ساخت و استقرار برنامه-های وب بدون نگرانی از زیرساختی است که برنامه بر رویش اجرا می-شود. برنامه-های این موتور، با زبان-های سطح بالا به ویژه جاوا و پایتون و در چارچوب GAE نوشته می-شوند. گوگل به منظور گسترش این نوع برنامه-ها یک سطح رایگان مشخص از سرویس را ارائه می-دهد و زمانی که برنامه از سطح مشخصی از بار پردازشی، ذخیره-سازی و پهنای باند شبکه فراتر رفت، آنگاه شارژها بر مبنای میزان استفاده محاسبه می-شود.

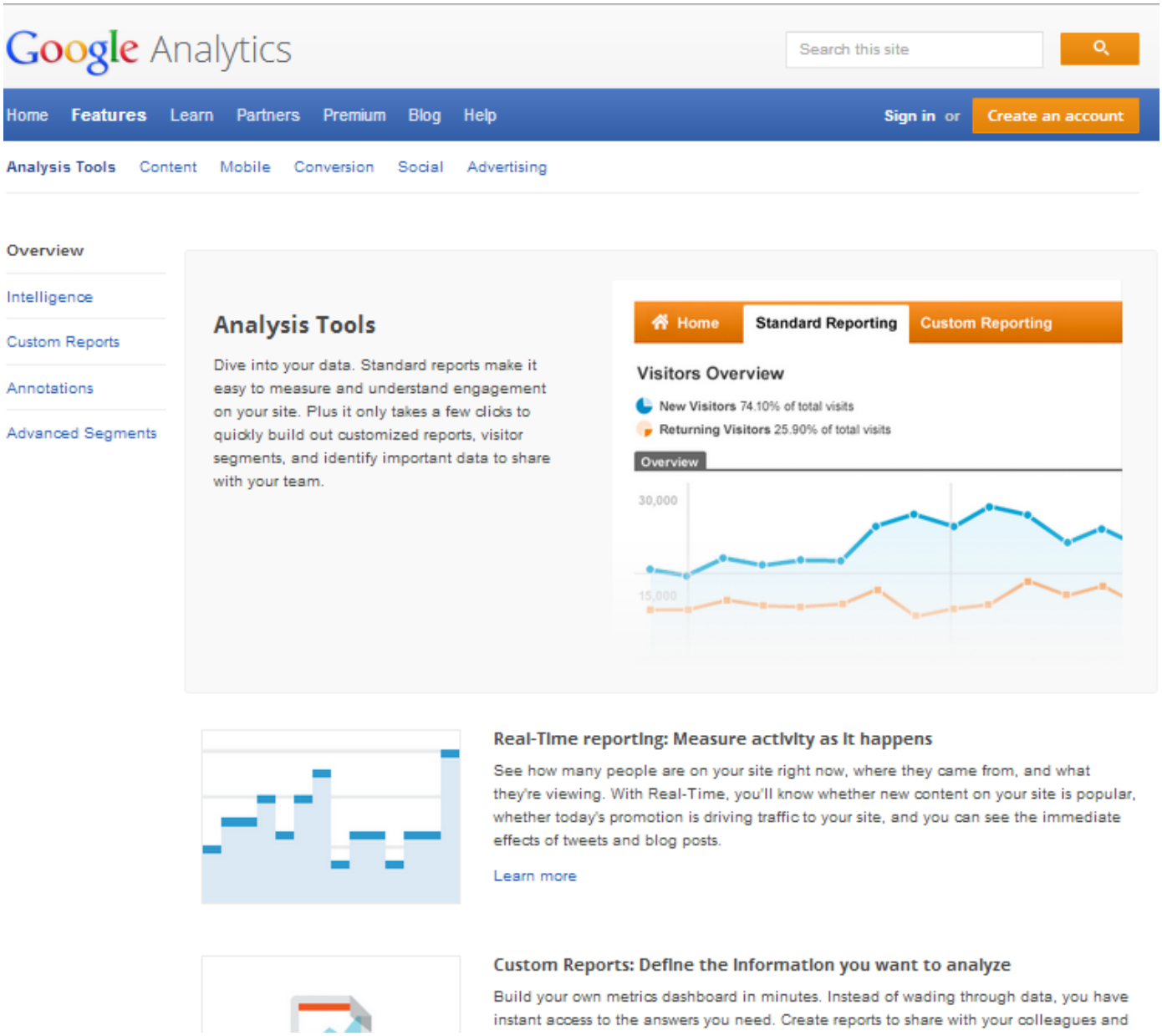
برنامه-های GAE را باید به گونه-ای نوشت که با زیرساخت گوگل وفق یابند. این مسئله، باعث محدودیت برنامه-های قابل اجرا در GAE می-گردد و علاوه بر آن، انتقال برنامه-ها به سکوی GAE و یا انتقال از این سکو به سایر سکو-های موجود دشوار می-شود.

از میان سرویس های ابری رایگان ارائه شده از سوی گوگل، به معرفی سرویس آنالیز گوگل بسنده کرده و تمرکز اصلی بر روی سکوی توسعه نرم‌افزاری این شرکت (GAE) می‌باشد.

Google Analytics

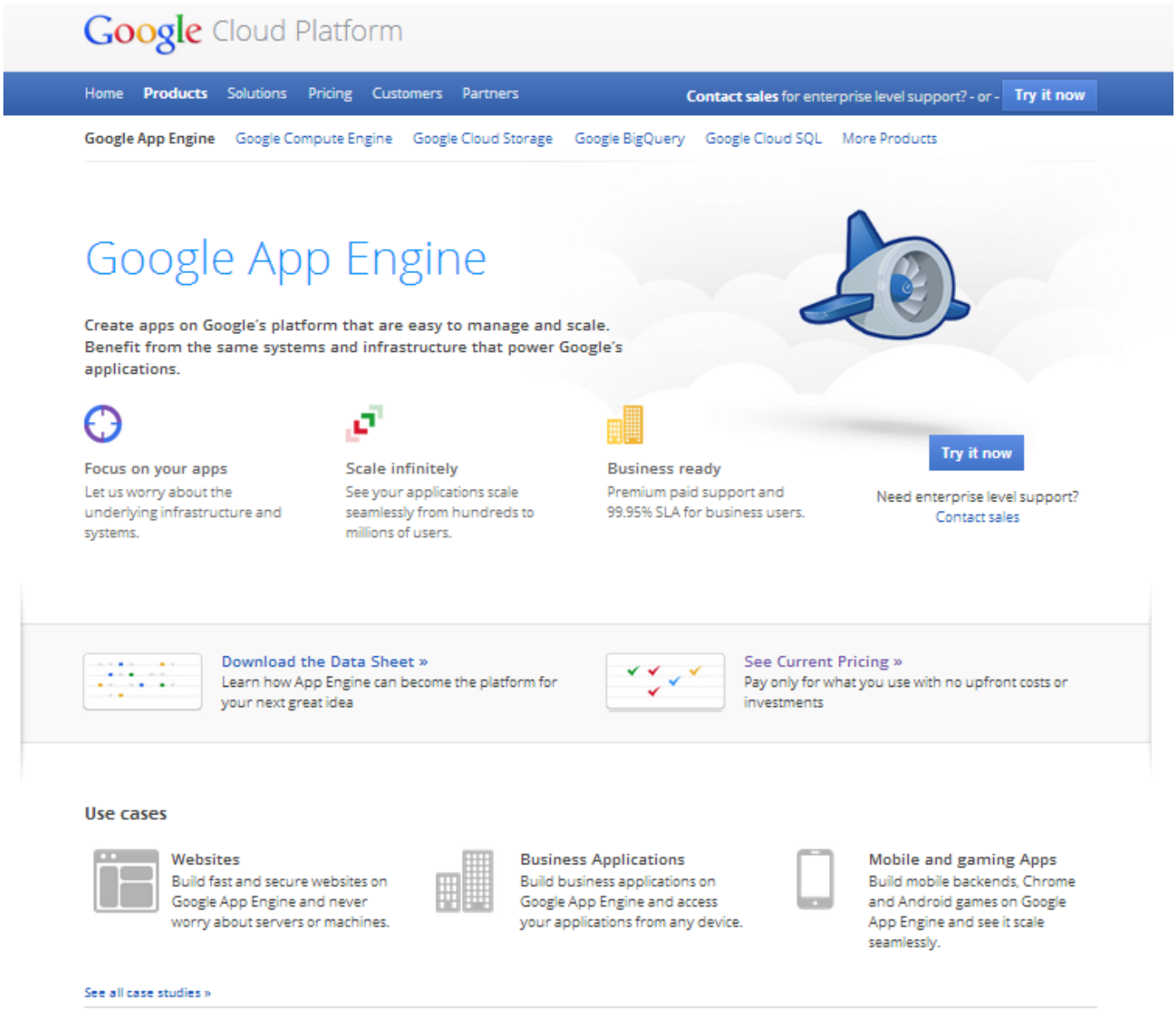
به اختصار GA نامیده می-شود و یک ابزار آماری است که تعداد و انواع بازدیدکنندگان وب-سایت و نحوه استفاده از وب-سایت را اندازه-گیری می-کند. این محصول بر روی بسته تحلیلی Urchin 5 که گوگل در سال 2006 آن را خریداری نمود، ساخته شده است. این سرویس رایگان عرضه می-شود و فرآیند تحلیل را با استفاده از یک قطعه کد جاوا اسکریپت به نام Google Analytics Tracking Code با پیاده‌سازی در تگ صفحه وب انجام می-شود.

این کد با اولین بارگذاری صفحه در سیستم- کاربران، به جمع آوری اطلاعات مورد نیاز پرداخته و برای پردازش به سرورهای GA باز پس می-فرستد. این کد با کمک Cookie مرورگر اطلاعات مورد نیاز را جمع آوری می-نماید.



آشنایی با Google App Engine

GAE یک سکو به عنوان سرویس می-باشد و مبتنی بر ابر گوگل است و بر روی زیر ساخت- گوگل مستقر شده است.



این سرویس توسعه دهندگان را قادر می-سازد تا برنامه-های وب ایجاد کرده و بر روی سرور-های گوگل مستقر سازند و گوگل مدیریت زیرساخت را بر عهده گیرد و اعمالی مانند نظارت، برطرف کردن اشکالات احتمالی، خوشه بندی، مدیریت وهله‌سازی ماشین‌های مجازی و غیره را انجام دهد. برای اجرای یک برنامه در GAE ابتدا باید استانداردهای سکوی گوگل رعایت شود. این استانداردها دامنه برنامه-هایی که قابل اجرا می-باشند را بسیار محدود می-نماید و قابلیت حمل آن-ها را کاهش می-دهد.

محدودیت-هایی که این سکو ایجاد می-کند، با خود مزایایی را به همراه می-آورد که در زیر به آن-ها اشاره می-گردد:

وب سرویس-های پویا بر مبنای استانداردهای رایج

توسعه خودکار و توازن بار بین ماشین‌های وهله‌سازی شده که مورد استفاده وب سرویس است.

اعتبارسنجی با استفاده از API موجود در گوگل.

فضای ذخیره سازی ماندگار با قابلیت جستجو، مرتب سازی و مدیریت تراکنش.

صف کاری و زمان بندی کاری

محیط توسعه سمت مشتری(توسعه دهندگان) برای شبیه سازی GAE در سیستم محلی.

پشتیبانی از محیط اجرا جاوا و پایتون.

هنگامی که یک برنامه در GAE مستقر گردید، با استفاده از نام دامنه دلخواه یا با استفاده از آدرس تجاری Google Apps قابل دستیابی است. موتور Google Apps در حال حاضر برنامه-هایی که در جاوا و پایتون نوشته شده است را پشتیبانی می-کند و علاوه بر آن از زبان-های ماشین مجازی جاوا و چندین چارچوب تحت وب پایتون که WSGI و CGI را پشتیبانی می-کنند نیز با محیط GAE سازگاری دارند.

برنامه-هایی که در GAE اجرا می-شوند از سیستم عامل مستقل هستند یا به گفته گوگل بر روی Sand Box اجرا می-شوند. این ویژگی GAE را قادر می-سازد، سیستم را بهینه کند تا تقاضاهای وب، با بار ترافیکی فعلی منطبق شوند. همچنین برنامه-ها را قادر می-سازد با امنیت بالاتری کار کنند، زیرا تنها می-توانند به کامپیوترهایی متصل شوند که آدرس‌های مشخصی دارند و سرویس-ها را با استفاده از پروتکل Http و یا Https از پورت-های شناخته شده پاسخ دهند. از طرف دیگر برنامه-ها نیز به این میزان محدود شده که تنها فایل-ها را بخوانند. آن‌ها حق نوشتن فایل به صورت مستقیم بر روی سیستم-ها را ندارند و برای دستیابی به داده، باید از ذخیره داده در Cache یا سرویس ماندگار دیگری استفاده نمایند.

GAE یک سیستم انبار داده توزیع شده دارد که از پرس و جوها و تراکنش-ها پشتیبانی می-نماید. این انبار داده غیر رابطه-ای است، اما اشیاء داده یا موجودیت-هایی که خصوصیات لازم را دارند، ذخیره می-نماید. به همین علت در پرس و جوها می-توان از فیلتر نوع موجودیت بهره برد و آن-ها را به ترتیب خصوصیات مرتب نمود.

در نهایت توجه به مدل قیمت‌گذاری گوگل قابل توجه است. گوگل برای تشویق توسعه دهندگان در نوشتن برنامه با استفاده از GAE ، استقرار و توسعه برنامه را تا میزان مشخصی از منابع رایگان کرده است و با عبور از مقدار رایگان باید هزینه را به ازای مصرف پرداخت نمود. بر اساس جدول ارائه شده در سایت شرکت گوگل به ازای تجاوز از میزان مصرف رایگان، سیستم هزینه گذاری بر اساس تعرفه‌های زیر، اقدام به محاسبه حق شارژ می‌نماید و بدیهی است برای آگاهی از آخرین تعرفه‌ها و کسب اطلاعات بیشتر، مراجعه به صفحه سایت شرکت گوگل توصیه می‌شود:

مبلغ به ازای هر یک ساعت استفاده از CPU معادل 0.08 دلار

داده ذخیره شده به ازای هر گیگابایت در ماه معادل 0.18 دلار

پهنای باند خروجی به ازای هر گیگابایت معادل 0.12 دلار

پهنای باند ورودی رایگان

هزینه دریافت هر ایمیل معادل 0.0001 دلار

به منظور ذخیره اطلاعات در منبع داده پایدار، از API استفاده می-گردد که به ازای تعداد تراکنش-هایی که تبادل می-گردد، هزینه پرداخت می-شود. از آنجایی که بنا به تعداد تبادلات و نوع حافظه پایداری که استفاده می-گردد، هزینه متغیر است، خواننده محترم برای رویت لیست مذکور به منبع ذکر شده، ارجاع داده می-شود.

منبع سهمیه	سهمیه پیش فرض رایگان به ازای هر برنامه
مصرف CPU	28 ساعت به ازای هر برنامه در روز

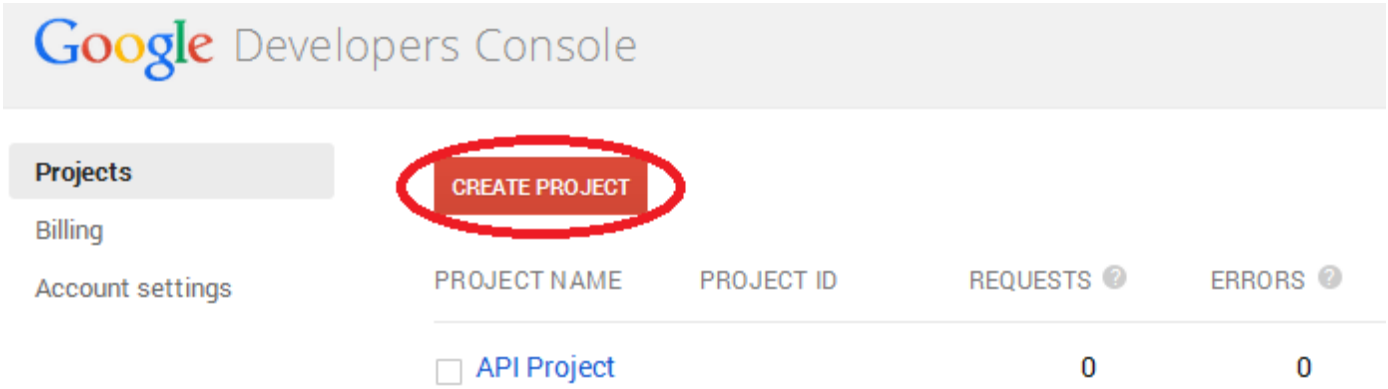
منبع سهمیه	سهمیه پیش فرض رایگان به ازای هر برنامه
منبع ذخیره پایدار داده	1 گیگابایت به ازای هر برنامه در ماه
پهنای باند ورودی	1 گیگابایت به ازای هر برنامه در روز
پهنای باند خروجی	1 گیگابایت به ازای هر برنامه در روز
تراکنش با منبع داده Datastore	50 هزار تراکنش برای خواندن و نوشتن به ازای هر برنامه در ماه
تراکنش با منبع داده Blobstore	5 گیگابایت به ازای هر برنامه در روز
ایمیل دریافتی	100 دریافت به ازای هر برنامه در روز

عنوان: **استفاده از نگارش سوم Google Analytics API در سرویس‌های ویندوز یا برنامه‌های وب**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۴ ۷:۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, google analytics

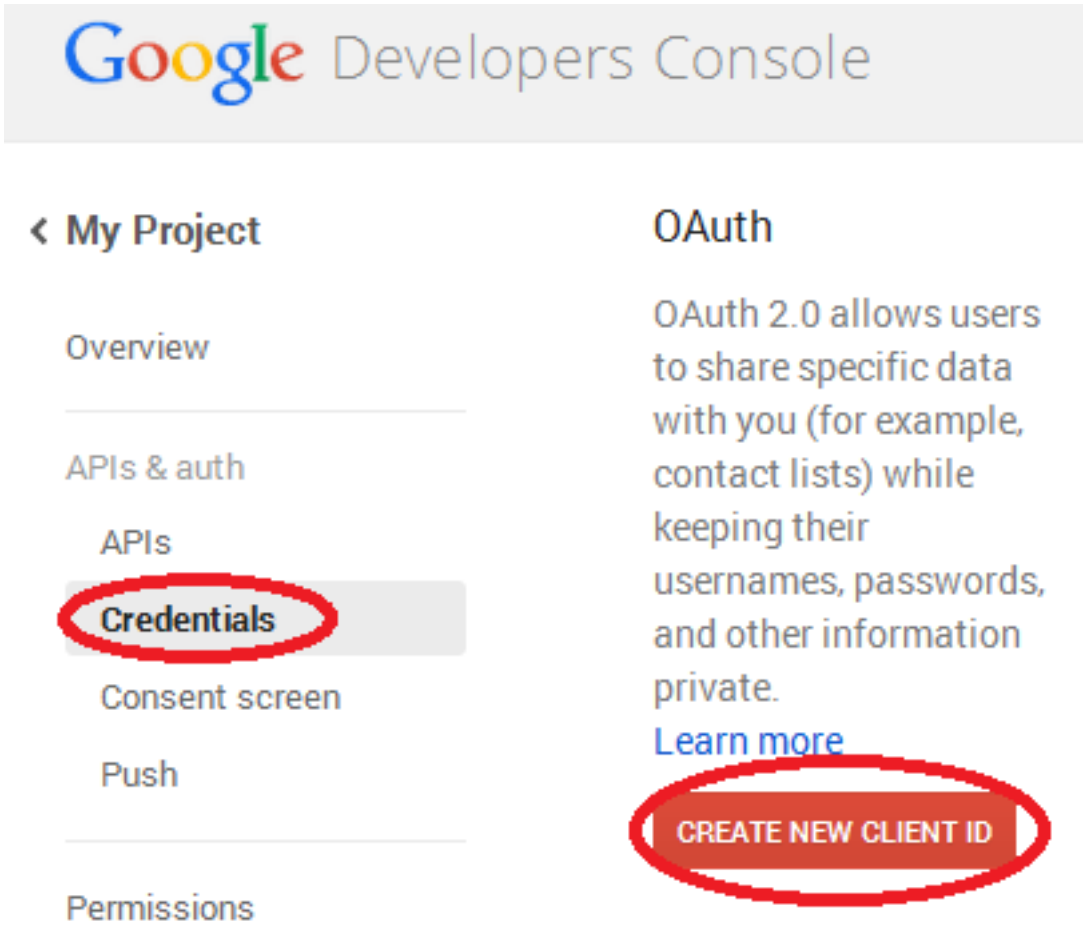
در زمان نگارش این مطلب، آخرین نگارش API مخصوص Google Analytics ، نگارش سوم آن است و ... کار کردن با آن دارای مراحل خاصی است که حتما باید رعایت شوند. در غیر اینصورت عملا در یک برنامه‌ی وب یا سرویس ویندوز قابل اجرا نخواهند بود. زیرا در حالت متداول کار با API مخصوص Google Analytics، ابتدا یک صفحه‌ی لاگین به Gmail باز می‌شود که باید به صورت اجباری، مراحل آن را انجام داد تا مشخصات تائید شده‌ی اکانت در حال استفاده‌ی از API، در پوشه‌ی AppData ویندوز برای استفاده‌های بعدی ذخیره شود. این مورد برای یک برنامه‌ی دسکتاپ معمولی مشکل ساز نیست؛ زیرا کاربر برنامه، به سادگی می‌تواند صفحه‌ی مرورگری را که باز شده‌است، دنبال کرده و به اکانت گوگل خود وارد شود. اما این مراحل را نمی‌توان در یک برنامه‌ی وب یا سرویس ویندوز پیگیری کرد، زیرا عموما امکان لاگین از راه دور به سرور و مدیریت صفحه‌ی لاگین به Gmail وجود ندارد یا بهتر است عنوان شود، بی‌معنا است. برای حل این مشکل، گوگل راه حل دیگری را تحت عنوان اکانت‌های سرویس، ارائه داده است که پس از ایجاد آن، یک یک فایل X509 Certificate برای اعتبارسنجی سرویس، در اختیار برنامه نویس قرار می‌گیرد تا بدون نیاز به لاگین دستی به Gmail، بتواند از API گوگل استفاده کند. در ادامه نحوه‌ی فعال سازی این قابلیت و استفاده از آن‌را بررسی خواهیم کرد.

ثبت برنامه‌ی خود در گوگل و انجام تنظیمات آن

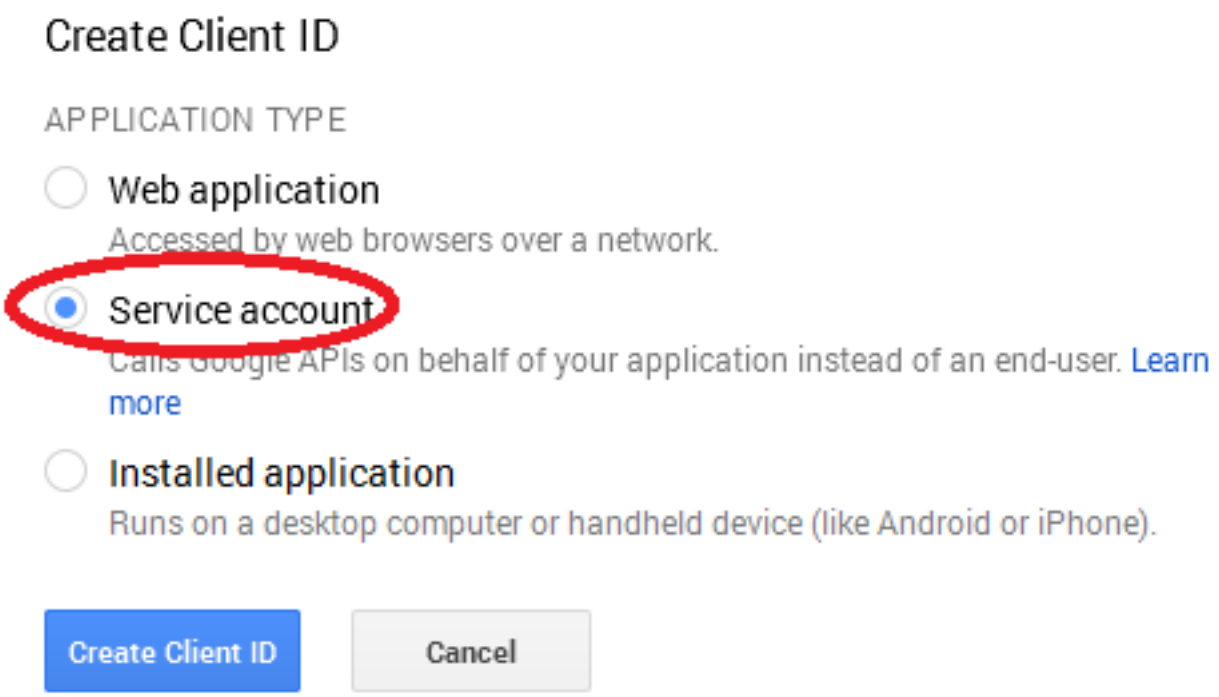
اولین کاری که برای استفاده از نگارش سوم Google Analytics API باید صورت گیرد، ثبت برنامه‌ی خود در Google Developer Console است. برای این منظور ابتدا به آدرس ذیل وارد شوید:
https://console.developers.google.com
سپس بر روی دکمه‌ی Create project کلیک کنید. نام دلخواهی را وارد کرده و در ادامه بر روی دکمه‌ی Create کلیک نمائید تا پروفایل این پروژه ایجاد شود.



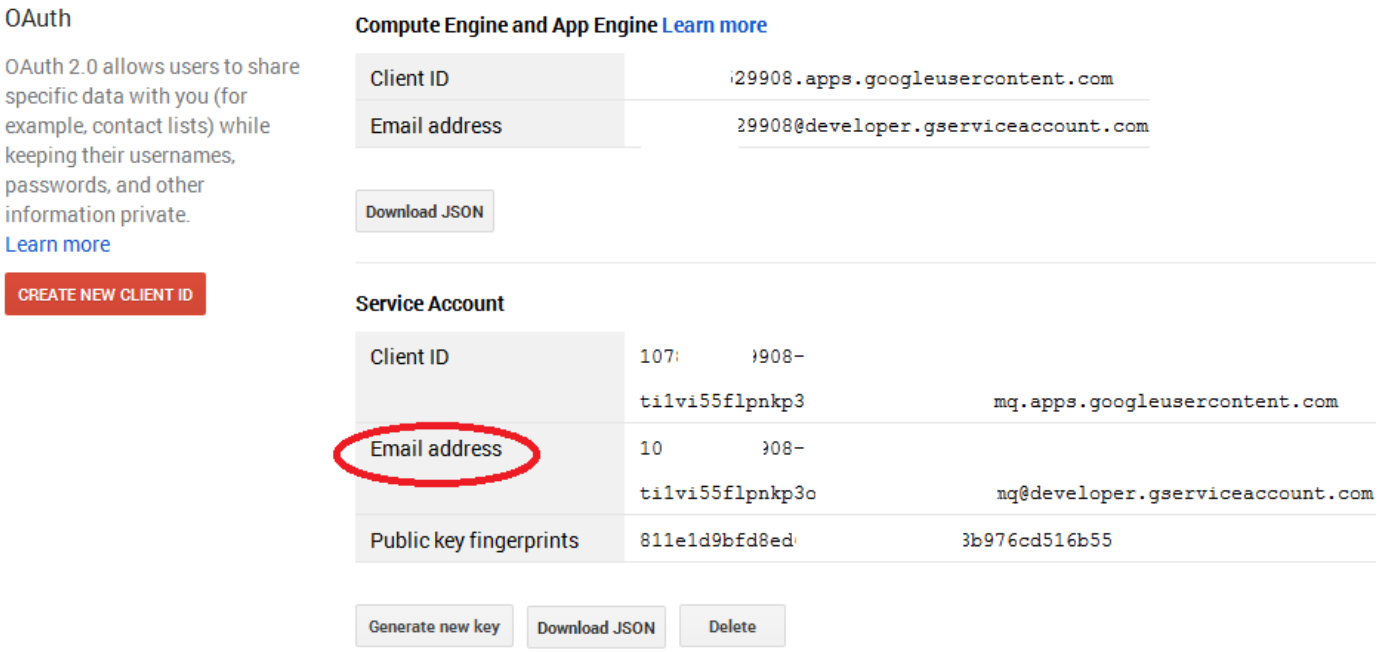
تنها نکته‌ی مهم این قسمت، بخاطر سپردن نام پروژه است. زیرا از آن جهت اتصال به API گوگل استفاده خواهد شد.
پس از ایجاد پروژه، به صفحه‌ی آن وارد شوید و از منوی سمت چپ صفحه، گزینه‌ی Credentials را انتخاب کنید. در ادامه در صفحه‌ی باز شده، بر روی دکمه‌ی Create new client id کلیک نمائید.



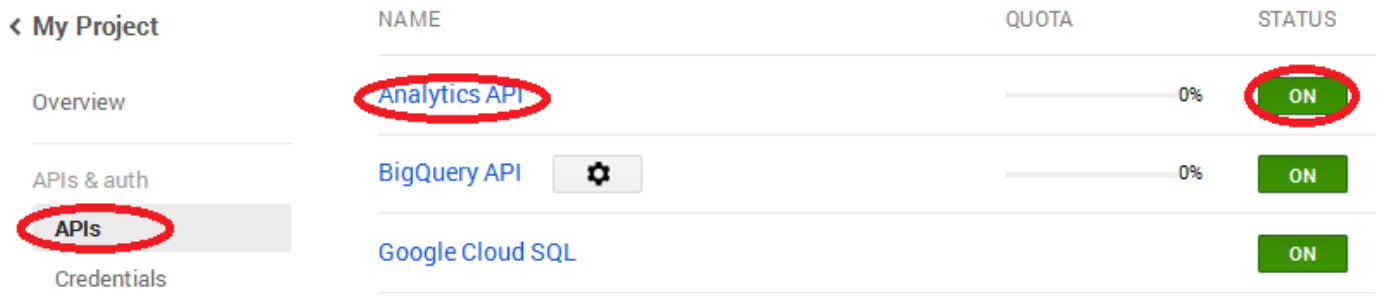
در صفحه‌ی باز شده، گزینه‌ی Service account را انتخاب کنید. اگر سایر گزینه‌ها را انتخاب نمائید، کاربری که قرار است از API استفاده کند، باید بتواند توسط مرورگر نصب شده‌ی بر روی کامپیوتر اجرا کننده‌ی برنامه، یکبار به گوگل لاگین نماید که این مورد مطلوب برنامه‌های وب و همچنین سرویس‌ها نیست.



در اینجا ابتدا یک فایل مجوز p12 را به صورت خودکار دریافت خواهید کرد و همچنین پس از ایجاد client id، نیاز است، ایمیل آن‌را جایی یادداشت نمائید:



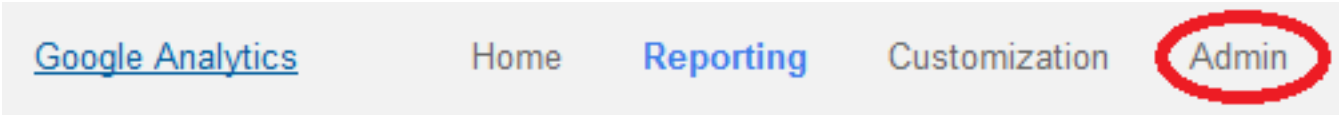
از این ایمیل و همچنین فایل p12 ارائه شده، جهت لاگین به سرور استفاده خواهد شد.
همچنین نیاز است تا به برگه‌ی APIs پروژه‌ی ایجاد شده رجوع کرد و گزینه‌ی Analytics API آن‌را فعال نمود:



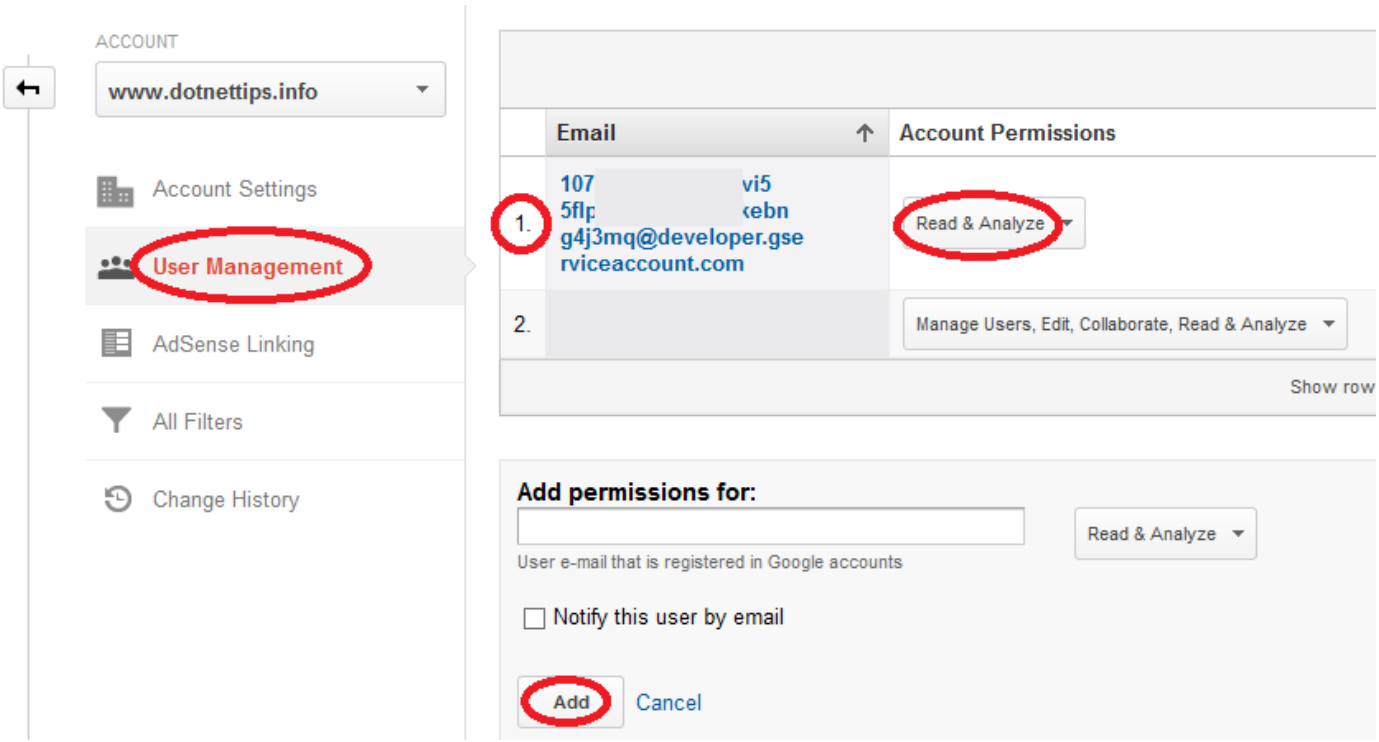
تا اینجا کار ثبت و فعال سازی برنامه‌ی خود در گوگل به پایان می‌رسد.

دادن دسترسی به Client ID ثبت شده در برنامه‌ی Google Analytics

پس از اینکه Client ID سرویس خود را ثبت کردید، نیاز است به اکانت Google Analytics خود وارد شوید. سپس در منوی آن، گزینه‌ی Admin را پیدا کرده و به آن قسمت، وارد شوید:



در ادامه به گزینه‌ی User management آن وارد شده و به ایمیل Client ID ایجاد شده در قسمت قبل، دسترسی خواندن و آنالیز را اعطاء کنید:



در صورت عدم رعایت این مساله، کلاینت API، قادر به دسترسی به Google Analytics نخواهد بود.

استفاده از نگارش سوم Google Analytics API در دات نت

قسمت مهم کار، تنظیمات فوق است که در صورت عدم رعایت آن‌ها، شاید نصف روزی را مشغول به دیباگ برنامه شوید. در ادامه نیاز است پیشنیازهای دسترسی به نگارش سوم Google Analytics API را نصب کنیم. برای این منظور، سه بسته‌ی نیوگت ذیل را توسط کنسول پاورشل نیوگت، به برنامه اضافه کنید:

```
PM> Install-Package Google.Apis
PM> Install-Package Google.Apis.auth
PM> Install-Package Google.Apis.Analytics.v3
```

پس از نصب، کلاس GoogleAnalyticsApiV3 زیر، جزئیات دسترسی به Google Analytics API را کپسوله می‌کند:

```
using System;
using System.Linq;
using System.Security.Cryptography.X509Certificates;
using Google.Apis.Analytics.v3;
using Google.Apis.Analytics.v3.Data;
using Google.Apis.Auth.OAuth2;
using Google.Apis.Services;

namespace GoogleAnalyticsAPIv3Tests
{
    public class AnalyticsQueryParameters
    {
        public DateTime Start { set; get; }
        public DateTime End { set; get; }
        public string Dimensions { set; get; }
        public string Filters { set; get; }
        public string Metrics { set; get; }
    }

    public class AnalyticsAuthentication
    {
        public Uri SiteUrl { set; get; }
        public string ApplicationName { set; get; }
        public string ServiceAccountEmail { set; get; }
        public string KeyFilePath { set; get; }
        public string KeyFilePassword { set; get; }

        public AnalyticsAuthentication()
        {
            KeyFilePassword = "notasecret";
        }
    }

    public class GoogleAnalyticsApiV3
    {
        public AnalyticsAuthentication Authentication { set; get; }
        public AnalyticsQueryParameters QueryParameters { set; get; }

        public GaData GetData()
        {
            var service = createAnalyticsService();
            var profile = getProfile(service);
            var query = service.Data.Ga.Get("ga:" + profile.Id,
                                QueryParameters.Start.ToString("yyyy-MM-dd"),
                                QueryParameters.End.ToString("yyyy-MM-dd"),
                                QueryParameters.Metrics);
            query.Dimensions = QueryParameters.Dimensions;
            query.Filters = QueryParameters.Filters;
            query.SamplingLevel = DataResource.GaResource.GetRequest.SamplingLevelEnum.HIGHERPRECISION;
            return query.Execute();
        }

        private AnalyticsService createAnalyticsService()
        {
            var certificate = new X509Certificate2(Authentication.KeyFilePath,
```

```
Authentication.KeyFilePassword, X509KeyStorageFlags.Exportable);
            var credential = new ServiceAccountCredential(
                new ServiceAccountCredential.Initializer(Authentication.ServiceAccountEmail)
                {
                    Scopes = new[] { AnalyticsService.Scope.AnalyticsReadonly }
                }.FromCertificate(certificate));

            return new AnalyticsService(new BaseClientService.Initializer
            {
                HttpClientInitializer = credential,
                ApplicationName = Authentication.ApplicationName
            });
        }

        private Profile getProfile(AnalyticsService service)
        {
            var accountListRequest = service.Management.Accounts.List();
            var accountList = accountListRequest.Execute();
            var site = Authentication.SiteUrl.Host.ToLowerInvariant();
            var account = accountList.Items.FirstOrDefault(x =>
x.Name.ToLowerInvariant().Contains(site));
            var webPropertyListRequest = service.Management.Webproperties.List(account.Id);
            var webPropertyList = webPropertyListRequest.Execute();
            var sitePropertyList = webPropertyList.Items.FirstOrDefault(a =>
a.Name.ToLowerInvariant().Contains(site));
            var profileListRequest = service.Management.Profiles.List(account.Id, sitePropertyList.Id);
            var profileList = profileListRequest.Execute();
            return profileList.Items.FirstOrDefault(a => a.Name.ToLowerInvariant().Contains(site));
        }
    }
}
```

در اینجا در ابتدا بر اساس فایل p12 ایی که از گوگل دریافت شد، یک X509Certificate2 ایجاد می‌شود. پسورد این فایل مساوی است با ثابت notasecret که در همان زمان تولید اکانت سرویس در گوگل، لحظه‌ای در صفحه نمایش داده خواهد شد. به کمک آن و همچنین ServiceAccountEmail ایمیلی که پیشتر به آن اشاره شد، می‌توان به AnalyticsService لاگین کرد. به این ترتیب به صورت خودکار می‌توان شماره پروفایل اکانت سایت خود را یافت و از آن در حین فراخوانی service.Data.Ga.Get استفاده کرد.

مثالی از نحوه استفاده از کلاس GoogleAnalyticsApiV3

در ادامه یک برنامه‌ی کنسول را ملاحظه می‌کنید که از کلاس GoogleAnalyticsApiV3 استفاده می‌کند:

```
using System;
using System.Collections.Generic;
using System.Linq;

namespace GoogleAnalyticsAPIv3Tests
{
    class Program
    {

        static void Main(string[] args)
        {
            var statistics = new GoogleAnalyticsApiV3
            {
                Authentication = new AnalyticsAuthentication
                {
                    ApplicationName = "My Project",
                    KeyFilePath = "811e1d9976cd516b55-privatekey.p12",
                    ServiceAccountEmail = "10152bng4j3mq@developer.gserviceaccount.com",
                    SiteUrl = new Uri("http://www.dotnettips.info/")
                },
                QueryParameters = new AnalyticsQueryParameters
                {
                    Start = DateTime.Now.AddDays(-7),
                    End = DateTime.Now,
                    Dimensions = "ga:date",
                    Filters = null,
                    Metrics = "ga:users,ga:sessions,ga:pageviews"
                }
            }.GetData();
```

```
            foreach (var result in statistics.TotalsForAllResults)
            {
                Console.WriteLine(result.Key + " -> total:" + result.Value);
            }
            Console.WriteLine();

            foreach (var row in statistics.ColumnHeaders)
            {
                Console.Write(row.Name + "\t");
            }
            Console.WriteLine();

            foreach (var row in statistics.Rows)
            {
                var rowItems = (List<string>)row;
                Console.WriteLine(rowItems.Aggregate((s1, s2) => s1 + "\t" + s2));
            }

            Console.ReadLine();
        }
    }
}
```

چند نکته

ApplicationName همان نام پروژه‌ای است که ابتدای کار، در گوگل ایجاد کردیم.

KeyFilePath مسیر فایل مجوز p12 ایی است که گوگل در حین ایجاد اکانت سرویس، در اختیار ما قرار می‌دهد.

ServiceAccountEmail آدرس ایمیل اکانت سرویس است که در قسمت ادمین Google Analytics به آن دسترسی دادیم.

SiteUrl آدرس سایت شما است که هم اکنون در Google Analytics دارای یک اکانت و پروفایل ثبت شده‌است.

توسط AnalyticsQueryParameters می‌توان نحوه‌ی کوئری گرفتن از Google Analytics را مشخص کرد. تاریخ شروع و پایان گزارش گیری در آن مشخص هستند. در مورد پارامترهایی مانند Dimensions و Metrics بهتر است به مرجع کامل آن در گوگل مراجعه نمائید:

Dimensions & Metrics Reference

برای نمونه در مثال فوق، تعداد کاربران، سشن‌های آن و همچنین تعداد بار مشاهده‌ی صفحات، گزارشگیری می‌شود.

برای مطالعه بیشتر

Using Google APIs in Windows Store Apps

How To Use Google Analytics From C# With OAuth

#Google Analytic’s API v3 with C

NET Library for Accessing and Querying Google Analytics V3 via Service Account.

#Google OAuth2 C

نظرات خوانندگان

نویسنده: ایلیا اکبری فرد
تاریخ: ۱۳۹۳/۰۲/۲۴ ۱۰:۲۸

سلام و تشکر از مقاله خوبتان.
به نظر شما بهتر است که یک اکانت در آنالیتیکس تعریف کنیم و تمام وبسایت‌های خودمان را به این اکانت اضافه کنیم یا اینکه برای هر وبسایت یک اکانت مجزا بسازیم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۴ ۱۰:۳۶

- برای متد getProfile نوشته شده تفاوتی نمی‌کند. چون تمام اکانت‌های موجود را جهت یافتن آدرس سایت مدنظر جستجو می‌کند تا شماره پروفایل آن‌را برای کوئری گرفتن استخراج کند.
+ اکانت Client ID ساخته شده، برای تمام سرویس‌های متفاوت گوگل کاربرد دارد؛ و منحصر به Google Analytics نیست. فقط باید در قسمت APIs آن پروژه، سرویس موردنظر را فعال کرد.
- همچنین مزیت این روش نسبت به پروتکل‌های قدیمی گوگل که از نام کاربری و کلمه‌ی عبور Gmail استفاده می‌کردند، عدم نیاز به ذخیره سازی اطلاعات حساس اکانت گوگل خود در فایل کانفیگ برنامه است. بنابراین نیازی نیست تا اکانت‌های متفاوت گوگلی را برای این‌کار خاص ایجاد کنید.

نویسنده: ایلیا اکبری فرد
تاریخ: ۱۳۹۳/۰۲/۲۴ ۱۲:۳۱

با سلام.
همه چیز بر روی کلاینت درست کار میکند ولی وقتی سایت را بر روی سرور آپلود میکنم خطای زیر روی می‌دهد:

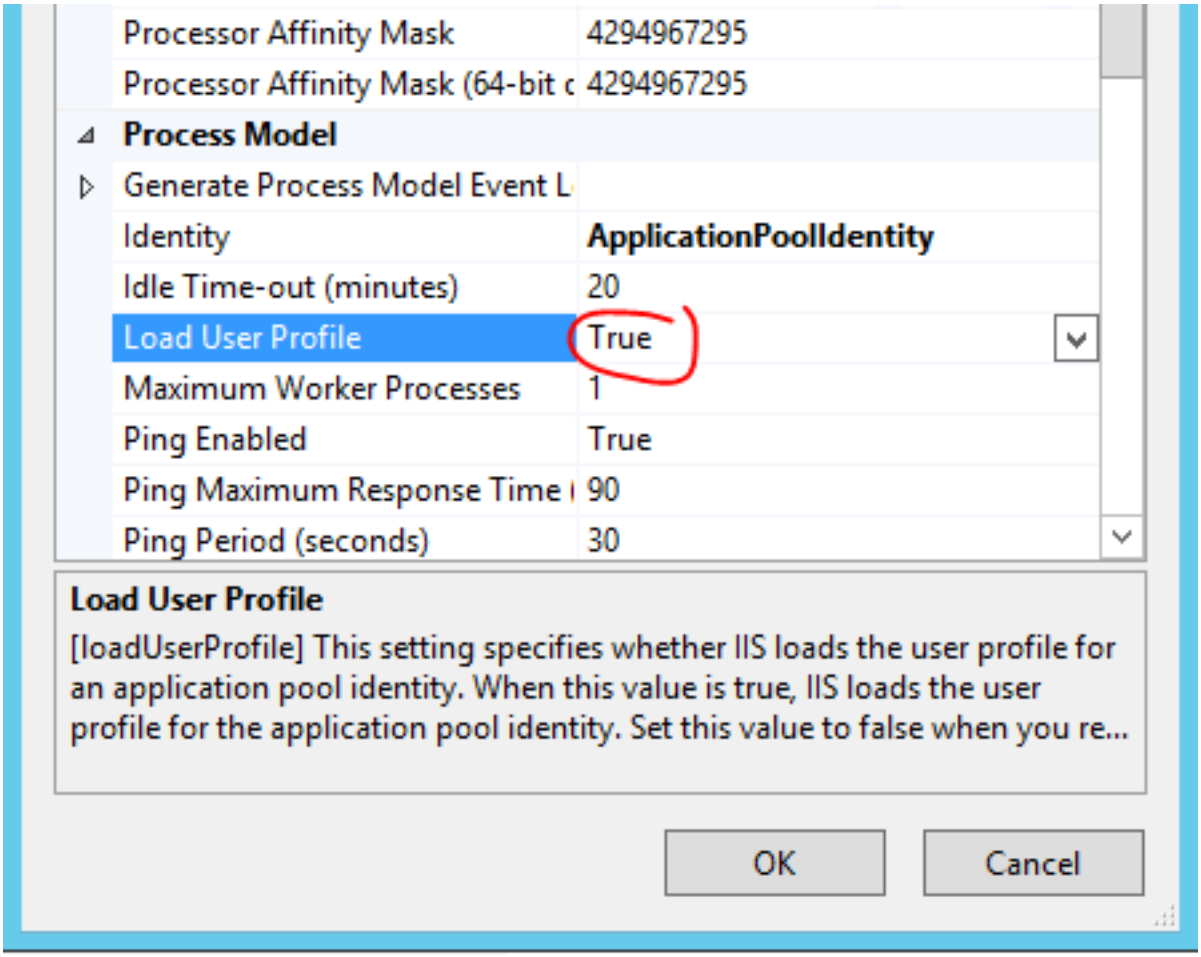
```
System.Security.Cryptography.CryptographicException: An internal error occurred
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۴ ۱۳:۲۷

این سه مورد را بررسی کنید:
- آیا مسیر فایل p12 را درست تنظیم کرده‌اید؟ در یک برنامه‌ی وب باید به این نحو باشد:

```
Server.MapPath("~/folder/" + filename)
```

- آیا در application pool برنامه‌ی شما load user profile فعال است؟ اگر خیر، crypto susbsystem کار نخواهد کرد:



- تنظیم زیر را هم امتحان کنید:

بجای

```
var certificate = new X509Certificate2(Authentication.KeyFilePath, Authentication.KeyFilePassword,
 X509KeyStorageFlags.Exportable);
```

بنویسید:

```
var certificate = new X509Certificate2(Authentication.KeyFilePath, Authentication.KeyFilePassword,
   X509KeyStorageFlags.MachineKeySet |
   X509KeyStorageFlags.PersistKeySet |
   X509KeyStorageFlags.Exportable);
```

نویسنده: ایلیا اکبری فرد
تاریخ: ۱۳۹۳/۰۲/۲۴ ۱۴:۲

حالت سوم پاسخ شما برای مورد من درست کار کرد. تشکر فراوان.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۹ ۱۵:۲۸

یک نکته‌ی تکمیلی

اگر کتابخانه‌ی Google.Apis.Analytics.v3 را بر روی یک سیستم دات نت 4 اجرا کنید، احتمالا خطای ذیل را دریافت خواهید کرد:

```
Could not load type 'System.Net.HttpStatusCode' from assembly System.Net
```

علت اینجا است که دات نت 4 نیاز به وصله‌ی KB2468871 دارد تا بتواند portable libraries را بارگذاری کند.

نویسنده: کامران
تاریخ: ۱۳۹۳/۰۵/۲۱ ۲:۵۲

از این ایراد میگیره میگه null هست، من اکانت ساختم پراپرتی هم درست کردم

```
Line 77: var webPropertyListRequest = service.Management.Webproperties.List(account.Id);

Object reference not set to an instance of an object.
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۰:۱۱

«من اکانت ساختم پراپرتی هم درست کردم»
کافی نیست. تمام مراحل باید انجام شوند.
این خطاها صرفا به معنای غیرمعتبر بودن یکی از مراحل است که سبب شده‌اند اطلاعات آن مرحله خاص قابل دریافت نباشد.

نویسنده: کامران
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۰:۵۳

نه اینکه فقط اکانت و پراپرتی درست کرده باشم، همه مراحل رو انجام دادم اما از خطی که خطا میده متوجه شدم که پرپرتی‌های من رو پیدا نمیکنه یعنی خالی برمیگردونه، قبلا هم کار کرده بودم همین مشکل پیش میومد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۱:۹

در متد getProfile :
- اگر account نال هست یعنی SiteUrl مشخص شده در ابتدای برنامه، در اکانت آنالیتیکس شما موجود نیست. روی سطر accountList.Items یک break-point قرار دهید تا لیست واقعی اکانت‌های ثبت شده را مشاهده کنید.
- اگر به مرحله‌ی اجرای accountList نمی‌رسید، یعنی ServiceAccountEmail معتبری وارد نشده‌است.

نویسنده: کامران
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۲:۳۲

متوجه اشکال شدم.
من کدهای شما رو کپی کردم و نخوندم چه کردید، کدها رو که بررسی کردم متوجه شدم شما توی لیست اکانتها که واکشی میکنید اکانتی که میخوایم باید همنام یا شبیه آدرس سایت(SiteUrl) باشه و طبیعتا من وقتی توی گوگل آنالیز اکانت و پروفایلهام رو میسازم نام دلخواه میدم، شما بجای استفاده از آی دی از اسم استفاده کردید که از روی اسم آی دی رو در بیارید برای همین چون اسم اکانت با آدرس سایتم یکی نبود چیزی پیدا نمیکرد :

```
var account = accountList.Items.FirstOrDefault(x => x.Name.ToLowerInvariant().Contains(site));
```

برای پراپرتی هم به همین صورت.

نویسنده: س محمدرضا برنتی
تاریخ: ۱۳۹۳/۱۱/۲۵ ۱۹:۲۰

با توجه به تحریم بودن این سرویس در ایران شما چه سرویس جایگزینی را پیشنهاد می‌کنید؟