Qiniu (Cloud) C# SDK

关于

此 C# SDK 适用于.NET Framework 2.0+，.NET Core 以及UWP(Windows 10 通用应用)，基于七牛云API参考手册构建。使用此 SDK 构建您的网络应用程序，能让您以非常便捷地方式将数据安全地存储到七牛云存储上。无论您的网络应用是一个网站程序，还是包括从云端（服务端程序）到终端（手持设备应用）的架构的服务或应用，通过七牛云存储及其 SDK，都能让您应用程序的终端用户高速上传和下载，同时也让您的服务端更加轻盈。

SDK文档

以下文档用于检索SDK接口、属性说明，它将有助于您理解SDK的结构。

- HTML在线浏览
- CHM文件下载

以下链接提供了一些示例(包含**UWP应用**示例)，参考这些示例可以帮助您更快熟悉如何使用这套SDK。

- qiniu/csharp-sdk/examples
- C# SDK使用指南 | 代码示例

如何安装

当前最新版本及最新改动同步在master分支。

以下是安装SDK的几种不同的方式，您可以根据需要来选择。

1. 直接添加DLL引用

您可以在这里找到所有的Release，选择您需要的版本下载，解压后将*.dll文件添加至项目引用。需要注意的是，此SDK依赖Json.NET，可以添加对应版本Newtonsoft.Json.dll引用或者使用NuGet来安装它：

```
Install-Package Newtonsoft.Json
```

2. 包管理器(NuGet)安装

或者从NuGet来安装，以Visual Studio 2013/2015为例，打开NuGet程序包管理器搜索 `Qiniu.Shared` 或者在控制台中键入以下命令：

```
Install-Package Qiniu.Shared
```

3. 从源码编译

当然，您也可以直接从源码编译

```
git clone https://github.com/qiniu/csharp-sdk
```

如何编译

推荐使用VS2013及以上版，根据目标平台选择对应的解决方案文件并打开：

目标	解决方案文件
.NET Framework 2.0	Qiniu.Net20.sln
.NET Framework 3.5	Qiniu.Net35.sln
.NET Framework 4.0	Qiniu.Net40.sln
.NET Framework 4.5	Qiniu.Net45.sln
.NET Framework 4.6	Qiniu.Net46.sln
.NET Core	Qiniu.Core.sln
Win10 UWP	Qiniu.UWP.sln

以上全部	Qiniu.ALL_VER.sln
单元测试(NUnit)	Qiniu.UnitTest.sln
单元测试(for UWP)	Qiniu.MSTest.sln

注意

如需编译 `Qiniu.Core.sln` （或 `Qiniu.UWP.sln` ），请先将 `Qiniu.NetCore` （或 `Qiniu.UWP` ）文件夹下的 `project.json` 和 `project.lock.json` 拷贝至 `Qiniu` 文件夹下，或者只拷贝 `project.json` 文件然后执行 `dotnet restore` 命令（推荐）。

编译其他版本时，如果 `Qiniu` 文件夹下有 `project.json` 或 `project.lock.json` ，请先删除。

API参考手册

- 对象存储API参考手册
- 数据处理API参考手册
- 融合CDN加速API参考手册

相关资源

如果您有任何关于我们文档或产品的建议和想法，欢迎到我们的技术论坛参与讨论。

- 技术论坛 - 在这里您可以和其他开发者愉快的讨论如何更好的使用七牛云服务
- 提交工单 - 如果您的问题不适合在论坛讨论或得不到回答，您也可以提交一个工单，技术支持人员会尽快回复您
- 博客 - 这里会持续发布市场活动和技术分享文章
- 微博
- 常见问题

贡献代码

1. Fork
2. 创建您的特性分支 git checkout -b my-new-feature
3. 提交您的改动 git commit -am ‘Added some feature’
4. 将您的修改记录提交到远程 git 仓库 git push origin my-new-feature
5. 然后到 github 网站的该 git 远程仓库的 my-new-feature 分支下发起 Pull Request

许可证

附录

速览.NET Core

下面是一个入门向导，如果您对如何创建及使用.NET Core程序还不太熟悉，可以参考；如果您已经比较熟悉，可以直接略过(或者也可以帮助我们改进^_^)。

开始之前，您需要准备 `dotnet` 工具，参见https://github.com/dotnet/cli/

以下步骤基本上都是在命令行终端下执行(如Windows的命令行控制台，Ubuntu/Mac的终端)。

1. 创建

首先切换到您的项目工作目录，然后键入以下命令来创建一个新的项目:

```
dotnet new
dotnet restore
```

注意

在执行 `dotnet restore` 之前，您可以在文本编辑器中打开并修改 `project.json` 文件(这个文件在执行 `dotnet new` 后就会自动生成)内容，下面是一个示例:

```
{
    "version": "1.0.0-*",
    "buildOptions": {
         "emitEntryPoint": true
        },

    "dependencies": {
        "Microsoft.NETCore.App": {
            "version": "1.0.1"
            },
            "Qiniu": "7.1.0.0",
            "Newtonsoft.Json": "9.0.1"
        },

        "frameworks": {
            "netcoreapp1.0": {
                "imports": "dnxcore50"
                }
        },

        "runtimes": {
            "win7-x64": {},
            "win7-x86": {},
            "osx.10.10-x64": {},
            "osx.10.11-x64": {},
            "ubuntu.14.04-x64": {},
            "ubuntu.16.04-x64": {}
        }
}
```

创建好项目之后就可以编写您的csharp代码了。

2. 发布

根据您的目标平台(操作系统)，选择其中一个来执行:

```
dotnet publish -r win7-x64
dotnet publish -r ubuntu.16.04-x64
dotnet publish -r osx.10.11-x64
```

发布之后，直接拷贝发布目录下的所有文件到目标计算机就可以直接使用。

如果目标平台上已经安装了.NET Core运行时，那么您需要做的就是编译：

```
dotnet build
```

3. 执行

如果您使用的是OSX 10.11(El Capitan), 请先安装openssl:

```
brew update
brew install openssl
brew link --force openssl
ln -s /usr/local/opt/openssl/lib/libcrypto.1.0.0.dylib /usr/local/lib/
ln -s /usr/local/opt/openssl/lib/libssl.1.0.0.dylib /usr/local/lib/
```

假设您编译好的程序(*built app*)是 `Example.dll` ，您可以键入以下命令来执行:

```
dotnet Example.dll
```

当然，执行这个命令之前， `dotnet` 工具是必须安装的。

如果您发布到Windows并且生成了exe文件，直接双击就可以运行。

速览Win10 UWP

Win10 UWP是指“Windows 10 通用应用”，它和之前的.NET应用开发有一些区别，比如文件存储使用的是StorageFolder和StorageFile等。具体请参阅Windows官方文档。

同样的，UWP解决方案中也包含一个project.json文件，以下是一个简单的示例：

```
{
  "dependencies": {
    "Microsoft.NETCore.UniversalWindowsPlatform": "5.1.0"
  },
  "frameworks": {
    "uap10.0": {}
  },
  "runtimes": {
    "win10-arm": {},
    "win10-arm-aot": {},
    "win10-x86": {},
    "win10-x86-aot": {},
    "win10-x64": {},
    "win10-x64-aot": {}
  }
}
```