官方文档地址https://docs.travis-ci.com/user/languages/csharp/

构建c# f#或vb项目

**本教程包括**

构建c#f#或vb项目，首先需要阅读入门和基本配置部分our[Getting started](https://docs.travis-ci.com/user/getting-started/) and [general build configuration](https://docs.travis-ci.com/user/customizing-the-build/)

**测试版本说明**

Travis CI support for C#, F#, and Visual Basic is currently in beta and may be removed or altered at any time. If you run into any problems, please report them in the [Travis CI issue tracker](https://github.com/travis-ci/travis-ci/issues/new) and cc @joshua-anderson @akoeplinger @nterry.

**构建环境**

目前Travis 构建c#f#和vb项目都是在linux上用mono实现的，这些是基于ECMA的c#标准和Clr标准的，可能没有微软的好

**概览**

设置项目如下:

language: csharp

solution: solution-name.sln

如果有了solution这个键值，系统就会用包管理器还原包并且构建解决方案，你可以检查脚本或是继续下一步

**脚本**

默认执行 xbuild /p:Configuration=Release solution-name.sln. xbuild是替换 MSBuild的 (Visual Studio的编译器). 想跳过就执行下面脚本

language: csharp

solution: solution-name.sln

script: # the following commands are just examples, use whatever your build process requires

- ./build.sh

- ./test.sh

- grep "Test Results" build.log

**NuGet 管理器**

默认执行 nuget restore solution-name.sln 安装时会扫描解决方案中引用的包。如果想跳过就执行下面脚本

language: csharp

solution: solution-name.sln

install:

- sudo apt-get install -y gtk-sharp2

- nuget restore solution-name.sln

**选择mono版本**

默认Travis用的是最新的mono，也能选择旧版本的mono，代码如下

language: csharp

mono:

- latest

- 3.12.0

- 3.10.0

...

Mono版本:

| **Version** | **Installed Packages** |
| --- | --- |
| 3.10.0 and later | mono-complete, mono-vbnc, fsharp, nuget, referenceassemblies-pcl |
| 3.8.0 | mono-complete, mono-vbnc, fsharp, nuget |
| 3.2.8 | mono-complete, mono-vbnc, fsharp |
| 2.10.8 | mono-complete, mono-vbnc |

**扩展**

[Coverity Scan](https://docs.travis-ci.com/user/coverity-scan/) 不能用，它只支持msbuild

**单元测试**

要用单元测试的话，先要安装测试的工具，NuGet里就有

运行单元测试的实例

**NUnit**[**#**](https://docs.travis-ci.com/user/languages/csharp/#NUnit)

language: csharp

solution: solution-name.sln

install:

- nuget restore solution-name.sln

- nuget install NUnit.Runners -Version 2.6.4 -OutputDirectory testrunner

script:

- xbuild /p:Configuration=Release solution-name.sln

- mono ./testrunner/NUnit.Runners.2.6.4/tools/nunit-console.exe ./MyPoject.Tests/bin/Release/MyProject.Tests.dll

**xunit**[**#**](https://docs.travis-ci.com/user/languages/csharp/#xunit)

language: csharp

solution: solution-name.sln

install:

- nuget restore solution-name.sln

- nuget install xunit.runners -Version 1.9.2 -OutputDirectory testrunner

script:

- xbuild /p:Configuration=Release solution-name.sln

- mono ./testrunner/xunit.runners.1.9.2/tools/xunit.console.clr4.exe ./MyPoject.Tests/bin/Release/MyProject.Tests.dll

*Note:* 有个bug，已经修复了

**解决方案级别的包管理器配置**

许多c#项目都有个 ` ./.nuget/packages.config `.

nuget 还原时也会还原这些引用包

language: csharp

solution: solution-name.sln

script:

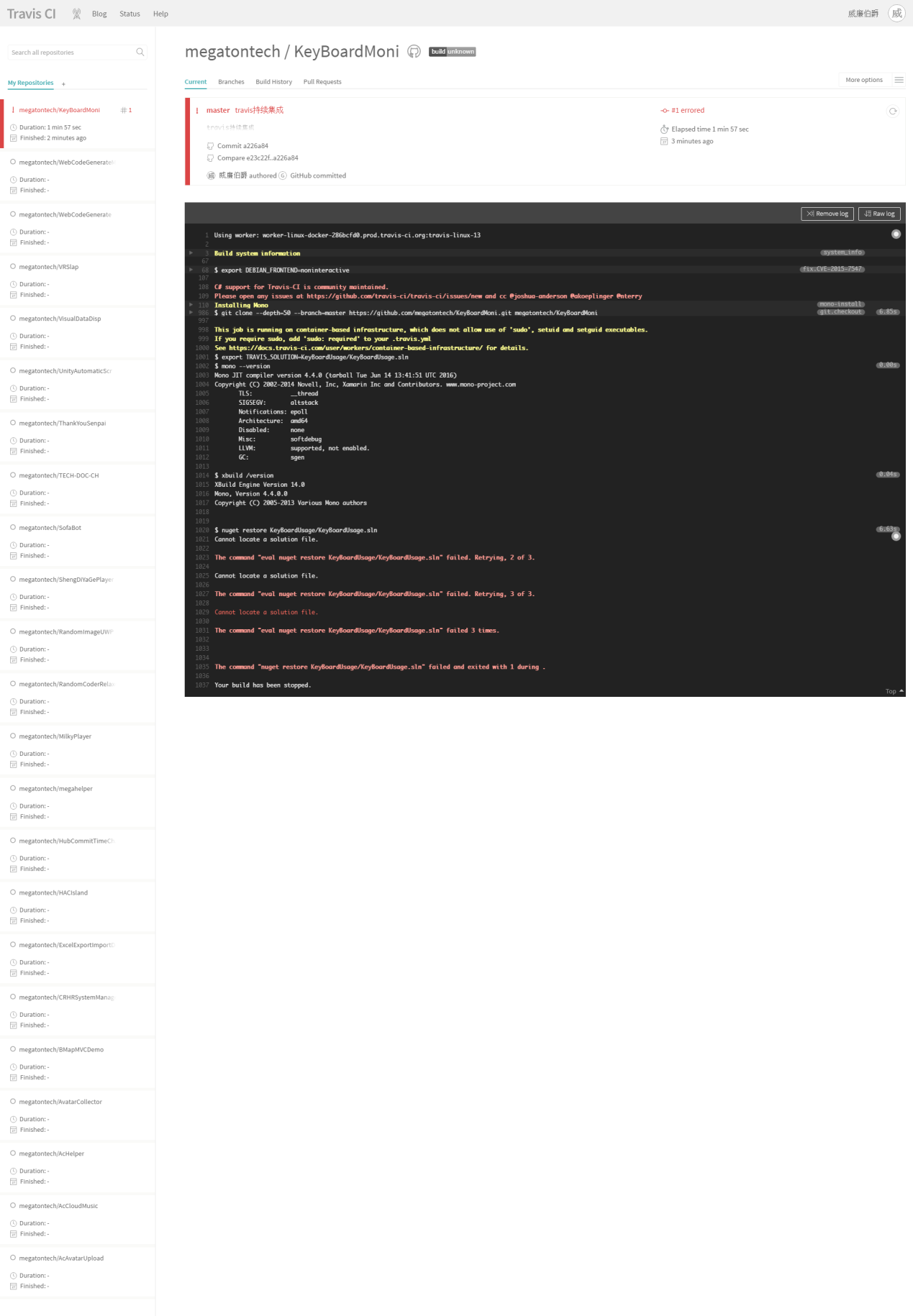
- xbuild /p:Configuration=Release solution-name.sln

- mono ./packages/xunit.runners.\*/tools/xunit.console.clr4.exe ./MyPoject.Tests/bin/Release/MyProject.Tests.dll

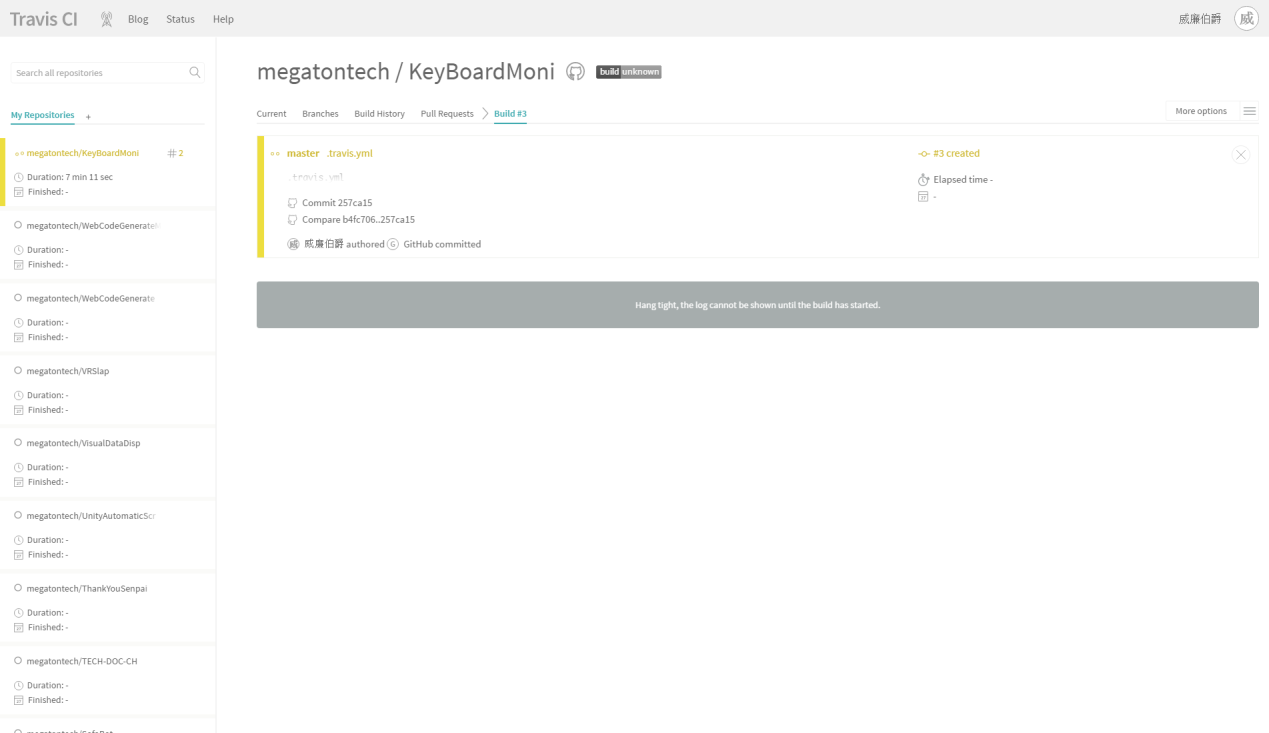
注意文中的扩展名带了星号，是为了避免测试用例中的硬编码

使用其他的测试框架也与此类似，，但是MSTest 不行的，它只能用在vs里

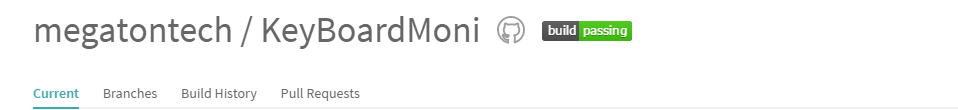
配置好呢就能看到效果了，比如



刚才的解决方案目录不对，换了之后呢正在构建

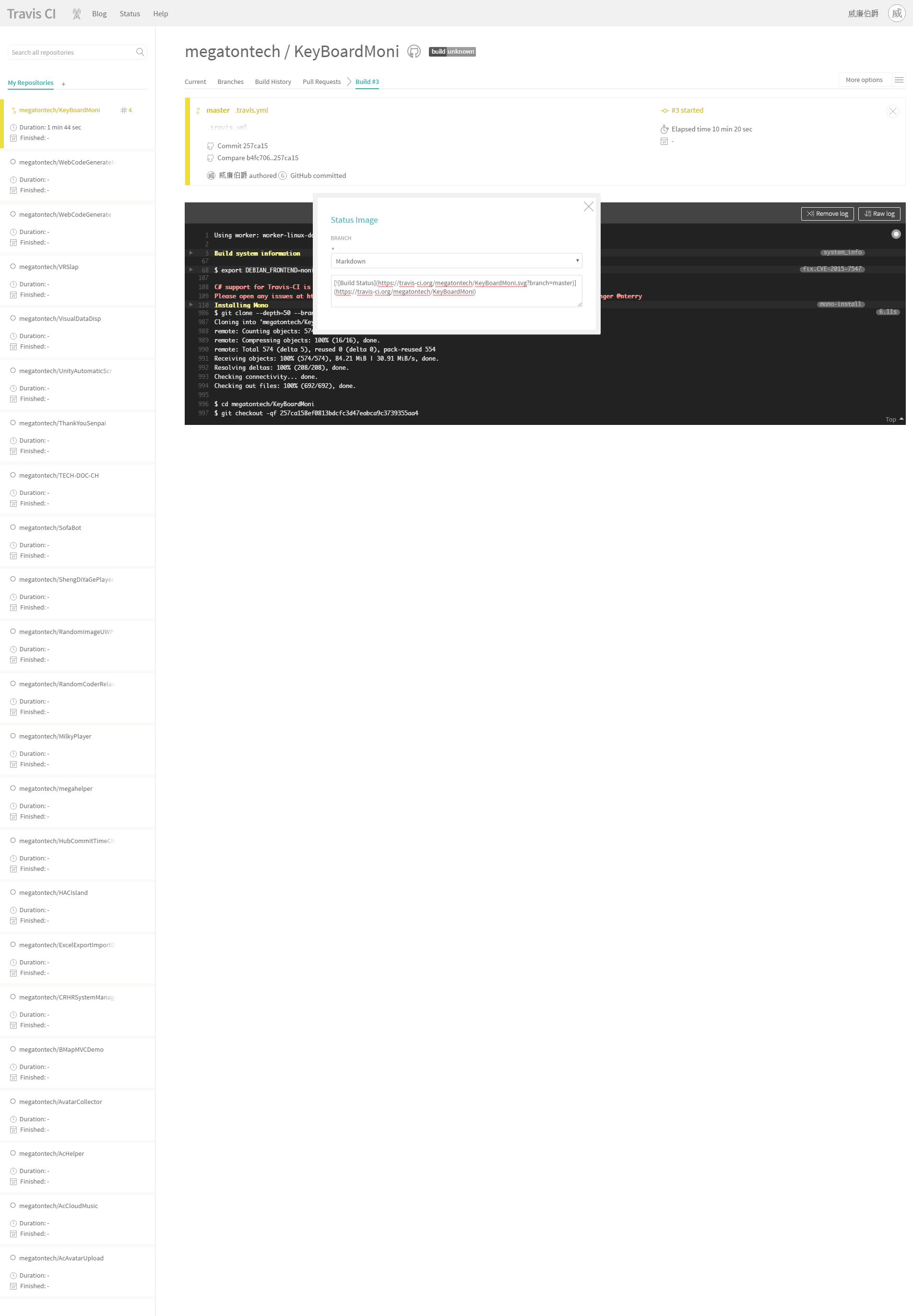


之后就通过了



在项目主页上显示构建状态只要很简单的操作

点击那个绿色的图标，会弹出一个对话框，里面选择markdown格式的直接粘到readme.md就行了



再刷新主页，已经能看到这个构建状态的标识了，是不是很简单

