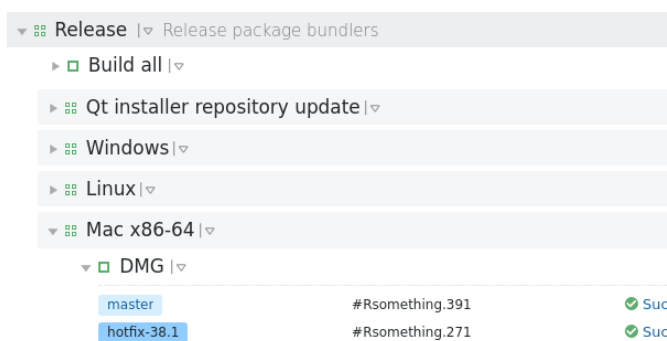


# MacOS: подписывание и нотаризация приложения UGENE

Для того, чтобы при скачивании и установке UGENE на MacOS, операционная система не ругалась и позволила установить UGENE приложение, необходимо подписать и нотаризовать приложение и поставляемый DMG-диск.

## Подписывание приложения UGENE

- 1) Строим приложение UGENE. В Teamcity есть соответствующий таск:



- 2) Перед тем, как подписать приложение, необходимо почистить код, т.к. некоторые external tools содержат multi-arch бинарные файлы, например i386 + x86\_64. В данной задаче для этого вызывается скрипт **codesign\_clear-i386.sh**

Этот скрипт проверяет все бинарные файлы и удаляет весь код, кроме x86\_64

Скрипт использует системную команду

- 3) Подписывание файлов в дереве приложения
  - Используем Apple Developer account: [appleid@alteametasoft.com](mailto:appleid@alteametasoft.com)  
В этом аккаунте созданы две подписи:
    - o Developer ID Application: Alteametasoft, S.L. (QBDVMPDVPQ)
    - o Developer ID Installer: Alteametasoft, S.L. (QBDVMPDVPQ)Подписи должны быть установлены в систему каждым юзером, который хочет иметь возможность подписывания
  - Используем системную команду **codesign** для подписывания. Подпись «Developer ID Application: Alteametasoft, S.L.». Скрипт **codesign.mac.sh** подписывает дерево приложения в следующей последовательности (вписок каталогов/файлов):
    - o Contents/Frameworks (все одной командой)
    - o Contents/Plugins/\* (все файлы, любого типа)
    - o Contents/Resources/\* (все файлы, любого типа)

- o Contents/MacOS/\*.dylib
  - o Contents/MacOS/\*.a
  - o Contents/MacOS/plugin\_checker
  - o Contents/MacOS/ugenem
  - o Contents/MacOS/ugenecl
  - o Contents/MacOS/ugeneui
- 4) Создаем подписанный pkg-файл приложения с помощью скрипта **productbuild.sh**. Скрипт использует **productbuild** для создания pkg-файла и **productsign** для подписания этого файла. Подпись «Developer ID Installer: Alteametasoftware, S.L.». PKG-файл – это установщик приложения.
  - 5) Создаем dmg-файл (монтируемый виртуальный диск) для хранения Samples, pkg-файл, readme.txt. Копируем их в произвольный каталог и с помощью системной команды **hdiutil** создаем dmg-файл и с помощью скрипта **codesign.mac.sh** подписываем этот файл.
  - 6) На этом подписывание завершено

## Нотаризация

Полуручная операция.

Нотаризация – это проверка Apple Gate Keeper нашего приложения, pkg-файла, dmg-диска на соответствие требованиям Apple. Проверка бывает двух видов:

- 1) Более требовательная проверка на размещение в Apple Store
- 2) Проверка приложений поставляемых через собственный канал разработчика (наш случай)

Для нотаризации необходимо отослать файл в Apple. Используем скрипт **notarize.sh**, который в свою очередь использует системную команду **altool**:

```
xcrun altool -notarize-app <options> <pkg-filename>
```

```
xcrun altool -notarize-app <options> <dmg-filename>
```

Результат выполнения команды уникальный UUID.

Нотаризация занимает некоторое время (иногда больше часа). Периодически проверяем результат с помощью **xcrun altool...**

Как только получили успешный результат нотаризации, то можно прикрепить этот результат к приложению и dmg-диску. Используем команду:

```
xcrun stapler staple <ugene-app>
```

```
xcrun stapler staple <dmg-filename>
```

## Рабочая последовательность процесса

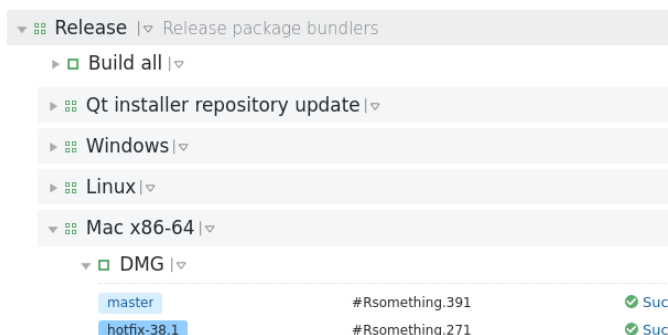
- ✓ строим приложение, получаем дерево файлов приложения
- ✓ чистим, бинарные файлы
- ✓ подписываем файлы
- ✓ создаем pkg-file
- ✓ отсылаем в Apple для нотаризации
- ✓ прикрепляем подпись к приложению
- ✓ создаем dmg-file
- ✓ отсылаем в Apple для нотаризации
- ✓ прикрепляем подпись к диску

# MacOS: signing and notarizing UGENE app

To ensure that when downloading and installing UGENE on MacOS, the operating system does not swear and allows you to install the UGENE application, you must sign and notarize the application and the supplied DMG disk.

## Signing the UGENE application

- 1) Build UGENE. There is a special task in Teamcity:



- 2) Before signing the application, you need to clean the code, because some external tools contain multi-arch binaries like i386 + x86\_64. In this task, the `codesign_clear-i386.sh` script is called for this. This script checks all binaries and removes all code except x86\_64
- 3) Signing files in the application tree
  - We use Apple Developer account: [appleid@alteametasoft.com](mailto:appleid@alteametasoft.com)

There are two signatures created in this account:

- o Developer ID Application: Alteametasoft, S.L. (QBDVMPDVPQ)
- o Developer ID Installer: Alteametasoft, S.L. (QBDVMPDVPQ)

Signatures must be installed in the system by every user who wants to be able to codesign

- We use system tool **codesign** for codesigning. Signature «Developer ID Application: Alteametasoft, S.L.». The **codesign.mac.sh** script signs the application tree in the following sequence (listing of directories / files):
  - o Contents/Frameworks
  - o Contents/Plugins/\*
  - o Contents/Resources/\*
  - o Contents/MacOS/\*.dylib
  - o Contents/MacOS/\*.a
  - o Contents/MacOS/plugin\_checker
  - o Contents/MacOS/ugenem
  - o Contents/MacOS/ugenecl
  - o Contents/MacOS/ugeneui
- 4) Create a signed application pkg file using the **productbuild.sh** script. The script uses **productbuild** to create a pkg file and **productsign** to sign this file. Signature "Developer ID Installer: Alteametasoft, S.L." PKG file is the installer for the application.
- 5) Create a dmg file (mounted virtual disk) for storing Samples, pkg file, readme.txt. Copy them to an arbitrary directory and use the hutil system command to create a dmg file and use the codesign.mac.sh script to sign this file.
- 6) This completes the signing

## Notarization

Semi-manual operation.

Notarization is checking the Apple Gate Keeper of our application, pkg file, dmg disk for compliance with Apple requirements. There are two types of verification:

- 1) More demanding verification for placement in the Apple Store
- 2) Checking applications delivered through the developer's own channel (our case)

For notarization, you need to send the file to Apple. We use the script **notarize.sh**, which in turn uses the system command **altool**:

```
xcrun altool -notarize-app <options> <pkg-filename>
```

```
xcrun altool -notarize-app <options> <dmg-filename>
```

The result of the command is a unique UUID.

Notarization takes some time (sometimes more than an hour). Check the result periodically with **xcrun altool....**

As soon as you have received a successful notarization result, you can attach this result to the application and dmg-disk. We use the command:

```
xcrun stapler staple <ugene-app>
```

```
xcrun stapler staple <dmg-filename>
```

## Working sequence of the process

build the application, get the application file tree

- ✓ clean, binaries
- ✓ sign files
- ✓ create pkg-file
- ✓ send to Apple for notarization
- ✓ attach a signature to the application
- ✓ create dmg-file
- ✓ send to Apple for notarization
- ✓ attach the signature to the disk