Making Customized Mozilla Installer file's Step-by-Step Guidance Documentation

Abstract

Creating a Mozilla Firefox installer for an open-source project involves several key steps to ensure a smooth and efficient process. The primary goal is to compile the Firefox source code and package it into an installer, which users can easily distribute and install.

First, you need to set up your development environment. This includes installing necessary tools such as Mercurial for version control, Python for scripting, and MozillaBuild for Windows or equivalent tools for Linux. Once the environment is ready, you clone the Firefox source code from the Mozilla Central repository.

Next, you configure the build system by creating a `mozconfig` file, which specifies the build options and paths. This step is crucial for customizing the build process to meet the specific needs of your project. After configuring, you run the build process using the `mach` command, which compiles the source code into executable binaries.

Finally, you package the compiled binaries into an installer. For Windows, this might involve using tools like NSIS (Null soft Scriptable Install System) to create a user-friendly installer. For Linux, you might create a DEB or RPM package depending on the target distribution.

Throughout the process, it is important to test the installer thoroughly to ensure it works correctly on all intended platforms. This includes verifying that the installation process is smooth, the browser runs without issues, and that you have handled all dependencies.

By following these steps, you can create a reliable and efficient Mozilla Firefox installer tailored to your open-source project's needs.

Now that you have enough background, we dive into steps to integrate Mozilla Central Source Files in order to make our customized installer file for "Persianfox" which I guess should be proper name for this project.

1st Step: Setting up Our Development Environment

In the first place, you need to install these required dependencies:

1.1. Download the latest release of Visual Studio's community edition, as shown in the figure1.1, including C++ Workload and "C++/CLI latest version support" enabled (see figures1.1.2 to 1.1.5)

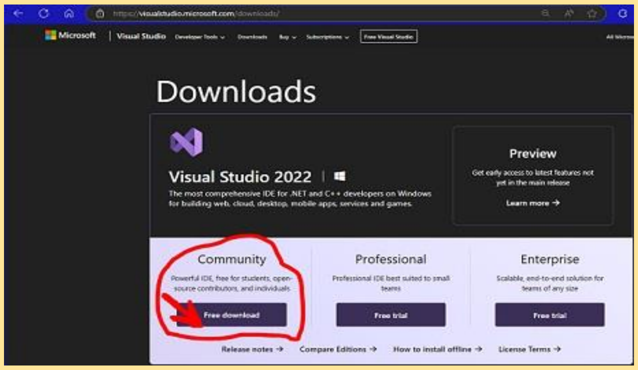

https://visualstudio.microsoft.com/downloads/
Microsoft | Visual Studio
Downloads
Visual Studio 2022
The most comprehensive IDE for .NET and C++ developers on Windows for building web, cloud, desktop, mobile apps, services and games.
Preview
Get early access to latest features not yet in the main release
Learn more →
Community
Powerful IDE, free for students, open-source contributors, and individuals
Free download
Professional
Professional IDE best suited to small teams
Free trial
Enterprise
Scalable, end-to-end solution for teams of any size
Free trial
Release notes →
Compare Editions →
How to install offline →
License Terms →

Figure 1.1

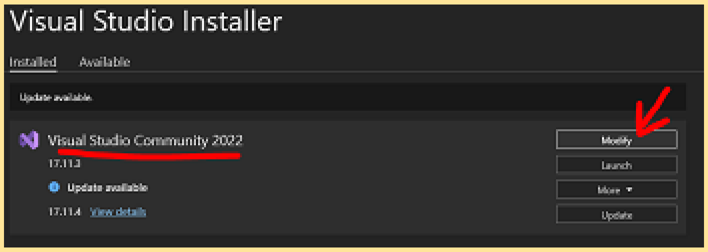
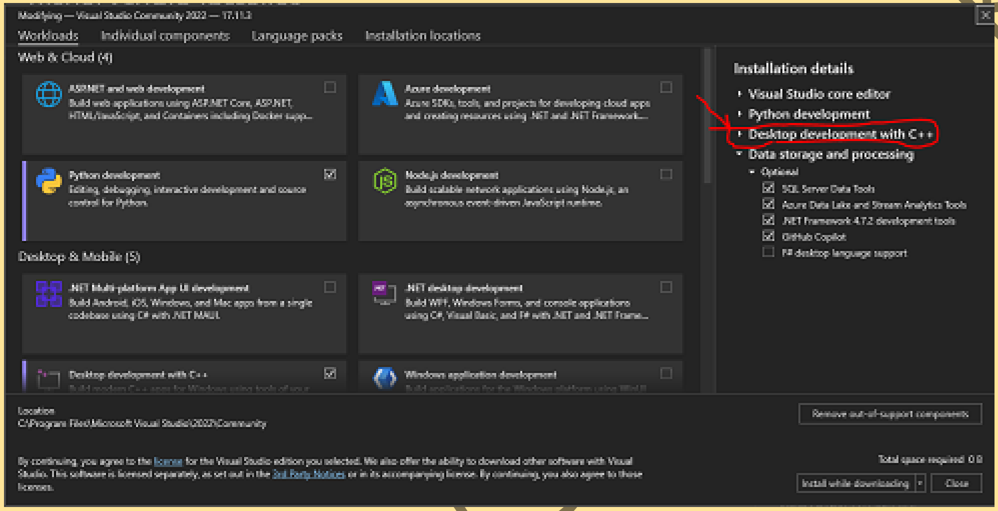


Figure 1.1.2


Modifying — Visual Studio Community 2022 — 17.11.3

Web & Cloud (4)

- ASP.NET and web development — Build web applications using ASP.NET Core, ASP.NET, HTML/JavaScript, and Containers including Docker supp...
- Azure development — Azure SDKs, tools, and projects for developing cloud apps and creating resources using .NET and .NET Framework...
- ✓ Python development — Editing, debugging, interactive development and source control for Python.
- Node.js development — Build scalable network applications using Node.js, an asynchronous event-driven JavaScript runtime.

Desktop & Mobile (5)

- .NET Multi-platform App UI development — Build Android, iOS, Windows, and Mac apps from a single codebase using C# with .NET MAUI.
- .NET desktop development — Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F# with .NET and .NET Frame...
- ✓ Desktop development with C++
- Windows application development

Installation details

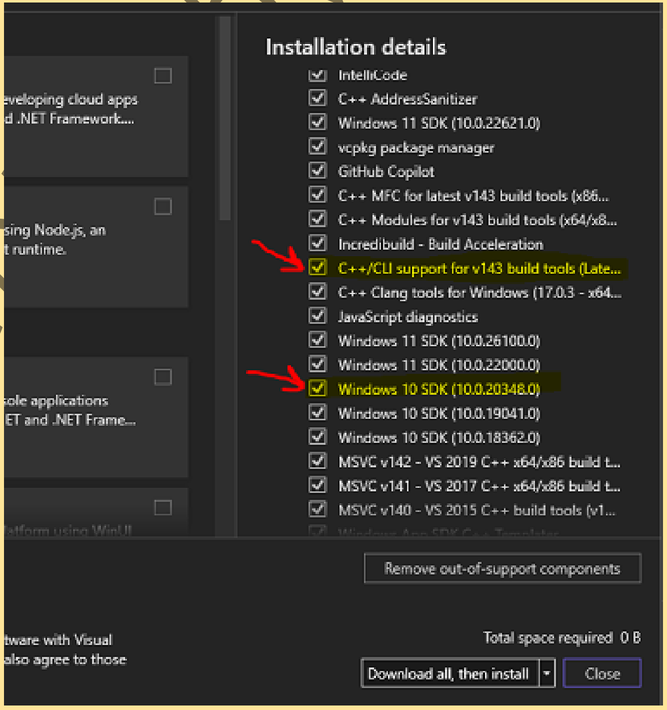
- Visual Studio core editor
- Python development
- Desktop development with C++
- Data storage and processing
  - Optional
    - ✓ SQL Server Data Tools
    - ✓ Azure Data Lake and Stream Analytics Tools
    - ✓ .NET Framework 4.7.2 development tools
    - ✓ GitHub Copilot
    - F# desktop language support

Location
C:\Program Files\Microsoft Visual Studio\2022\Community

By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

Total space required 0 B

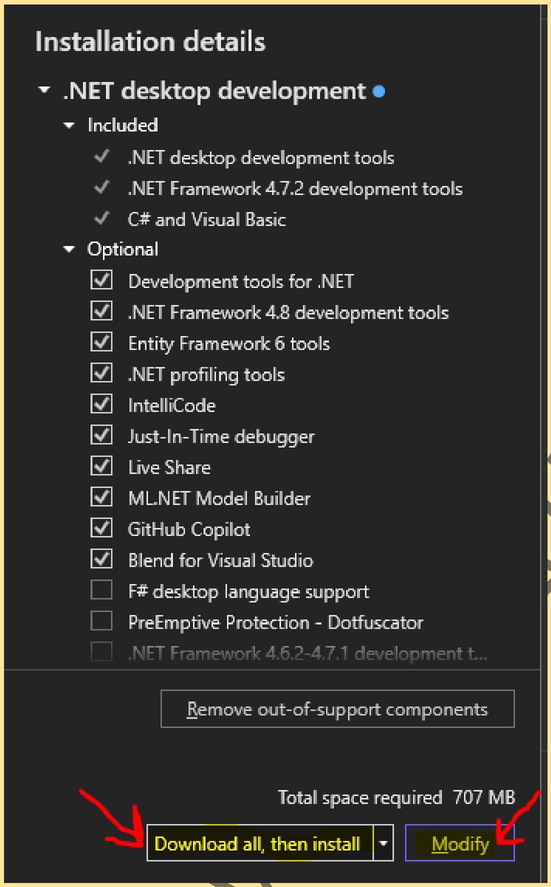
Figure 1.1.3


Installation details

- ✓ IntelliCode
- ✓ C++ AddressSanitizer
- ✓ Windows 11 SDK (10.0.22621.0)
- ✓ vcpkg package manager
- ✓ GitHub Copilot
- ✓ C++ MFC for latest v143 build tools (x86...
- ✓ C++ Modules for v143 build tools (x64/x8...
- ✓ Incredibuild - Build Acceleration
- ✓ C++/CLI support for v143 build tools (Late...
- ✓ C++ Clang tools for Windows (17.0.3 - x64...
- ✓ JavaScript diagnostics
- ✓ Windows 11 SDK (10.0.26100.0)
- ✓ Windows 11 SDK (10.0.22000.0)
- ✓ Windows 10 SDK (10.0.20348.0)
- ✓ Windows 10 SDK (10.0.19041.0)
- ✓ Windows 10 SDK (10.0.18362.0)
- ✓ MSVC v142 - VS 2019 C++ x64/x86 build t...
- ✓ MSVC v141 - VS 2017 C++ x64/x86 build t...
- ✓ MSVC v140 - VS 2015 C++ build tools (v1...

Total space required 0 B

Figure 1.1.4: Check **Win.10 SDK** as well, since we will need it for bootstrap environment


Installation details

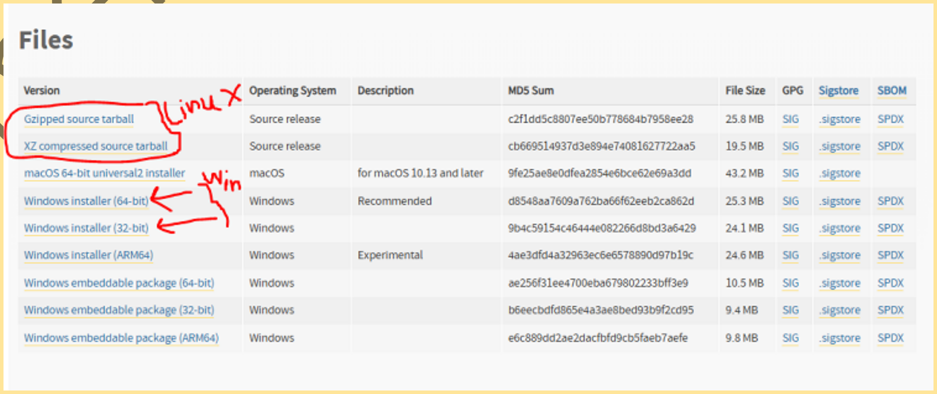
- .NET desktop development
  - Included
    - ✓ .NET desktop development tools
    - ✓ .NET Framework 4.7.2 development tools
    - ✓ C# and Visual Basic
  - Optional
    - [✓] Development tools for .NET
    - [✓] .NET Framework 4.8 development tools
    - [✓] Entity Framework 6 tools
    - [✓] .NET profiling tools
    - [✓] IntelliCode
    - [✓] Just-In-Time debugger
    - [✓] Live Share
    - [✓] ML.NET Model Builder
    - [✓] GitHub Copilot
    - [✓] Blend for Visual Studio
    - [ ] F# desktop language support
    - [ ] PreEmptive Protection - Dotfuscator
    - [ ] .NET Framework 4.6.2-4.7.1 development t...

Total space required 707 MB

Figure 1.1.5: Between two install / download options, choose "Download all, then install" option, as it assures optimized downloading time and lighter workload on CPU.

- Note that after modifying Visual Studio, you need to restart that and while installation you cannot use it

1.2. Download, install and deploy Python3 on your system, since Mozilla's build system relies on Python dependencies, so based on your system's architecture, you should use one the following installers from bottom of the page as shown in the figure 1.2


Files

Version	Operating System	Description	MD5 Sum	File Size	GPG	Sigstore	SBOM
Gzipped source tarball	Source release		c2f1dd5c8807ee50b778684b7958ee28	25.8 MB	SIG	.sigstore	SPDX
XZ compressed source tarball	Source release		cb669514937d3e894e74081627722aa5	19.5 MB	SIG	.sigstore	SPDX
macOS 64-bit universal2 installer	macOS	for macOS 10.13 and later	9fe25ae8e0dfea2854e6bce62e69a3dd	43.2 MB	SIG	.sigstore	
Windows installer (64-bit)	Windows	Recommended	d8548aa7609a762ba66f62eeb2ca862d	25.3 MB	SIG	.sigstore	SPDX
Windows installer (32-bit)	Windows		9b4c59154c46444e082266d8bd3a6429	24.1 MB	SIG	.sigstore	SPDX
Windows installer (ARM64)	Windows	Experimental	4ae3dfd4a32963ec6e6578890d97b19c	24.6 MB	SIG	.sigstore	SPDX
Windows embeddable package (64-bit)	Windows		ae256f31ee4700eba679802233bff3e9	10.5 MB	SIG	.sigstore	SPDX
Windows embeddable package (32-bit)	Windows		b6eecbdfd865e4a3ae8bed93b9f2cd95	9.4 MB	SIG	.sigstore	SPDX
Windows embeddable package (ARM64)	Windows		e6c889dd2ae2dacfbfd9cb5faeb7aefe	9.8 MB	SIG	.sigstore	SPDX

Figure 1.2. Choose installer based on your system type. If you do not know it yet, see **figures 1.2.1** and **1.2.2**, to determine your sys. Type.

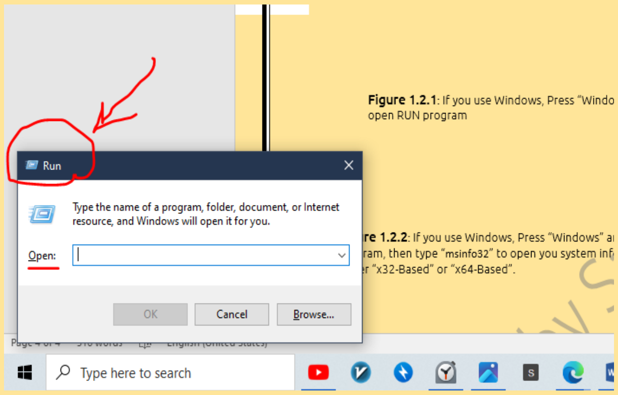
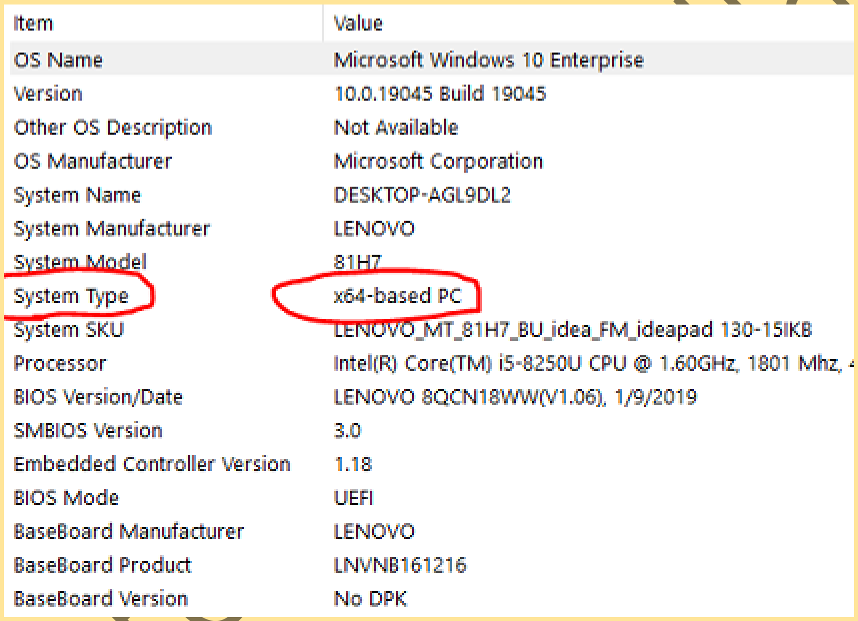

Run
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.
Open:
OK
Cancel
Browse...
Type here to search

Figure 1.2.1: If you use Windows, Press "Windows" and "R" key at the same time to open RUN program. Then type "**msinfo32**" to open you system info.



Item	Value
OS Name	Microsoft Windows 10 Enterprise
Version	10.0.19045 Build 19045
Other OS Description	Not Available
OS Manufacturer	Microsoft Corporation
System Name	DESKTOP-AGL9DL2
System Manufacturer	LENOVO
System Model	81H7
System Type	x64-based PC
System SKU	LENOVO_MT_81H7_BU_idea_FM_ideapad 130-15IKB
Processor	Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz, 1801 Mhz,
BIOS Version/Date	LENOVO 8QCN18WW(V1.06), 1/9/2019
SMBIOS Version	3.0
Embedded Controller Version	1.18
BIOS Mode	UEFI
BaseBoard Manufacturer	LENOVO
BaseBoard Product	LNVNB161216
BaseBoard Version	No DPK

Figure 1.2.2: There you see System Type, which can be either "x32-Based" or "x64-Based".

1.3. Download and install MozillaBuild packages from https://wiki.mozilla.org/MozillaBuild

1,3,1, Clone the Mozilla Central Repository, by typing the below bash command in Windows CMD; so that you don't have to download raw C++ sources separately, (I mean: .cpp files in https://searchfox.org/mozilla-central/source/)

```bash
hg clone https://hg.mozilla.org/moz
```

➢ Note that if CMD cannot recognize "hg" command, it is due to uninstalled Tortoise Hg, so you need to install it, as I show you in the next page.

1,3,2. How to install and configure Tortoise Hg?

1. 1st Step: Go to https://tortoisehg.bitbucket.io/download/index.html , now you must see this web page 👇

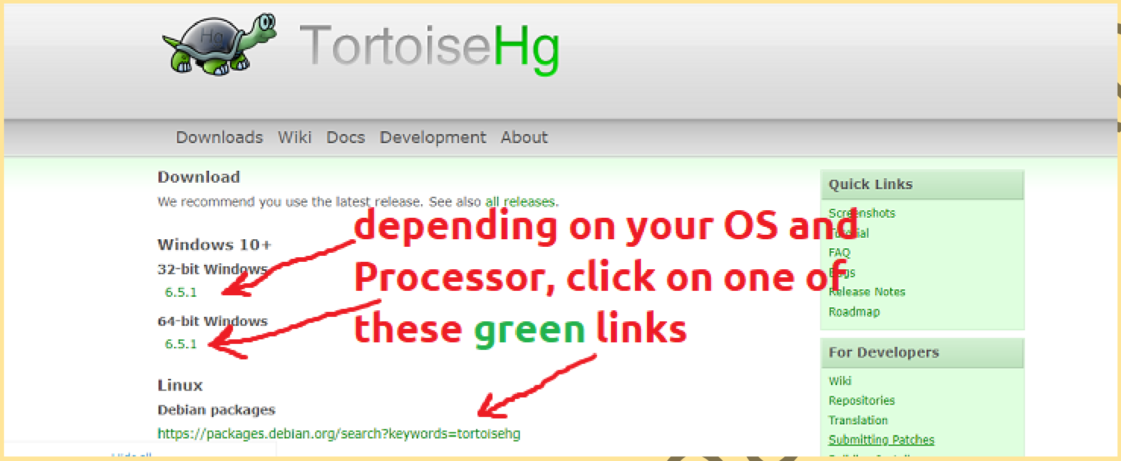
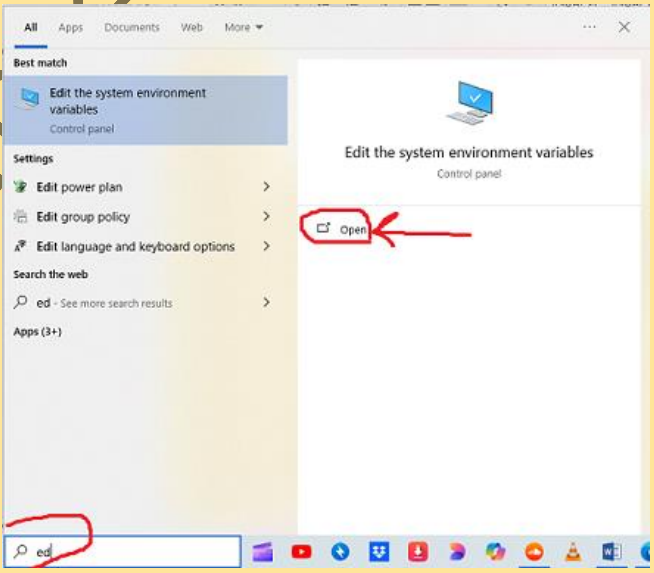

TortoiseHg
Downloads Wiki Docs Development About
Download
We recommend you use the latest release. See also all releases.
Windows 10+
32-bit Windows
6.5.1
64-bit Windows
6.5.1
Linux
Debian packages
https://packages.debian.org/search?keywords=tortoisehg
Quick Links
Screenshots
Tutorial
FAQ
Bugs
Release Notes
Roadmap
For Developers
Wiki
Repositories
Translation
Submitting Patches
depending on your OS and Processor, click on one of these green links

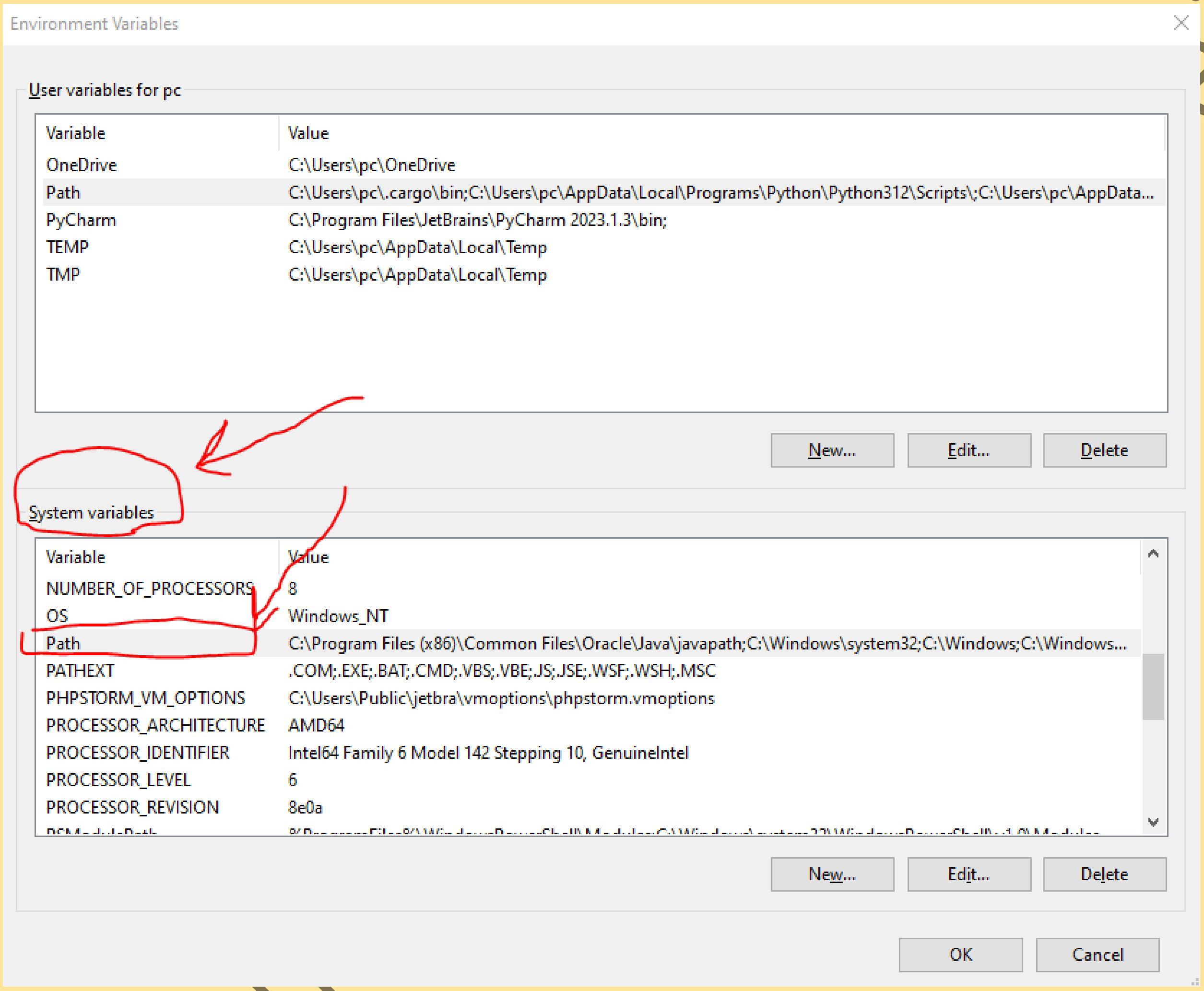
Figure 1.3.2: Click on of these links as mentioned in the image; if you are on Windows, find out your system type as guided in the previous section

2. 2nd Step: Add TortoiseHg path to System Environment Variable:

   2.1. Type "Edit the system environment variables" in your window's search bar, and open.


All Apps Documents Web More
Best match
Edit the system environment variables
Control panel
Settings
Edit power plan
Edit group policy
Edit language and keyboard options
Search the web
ed - See more search results
Apps (3+)
Edit the system environment variables
Control panel
Open
ed

2.2. Navigate to the System variables, and find "Path" variables.



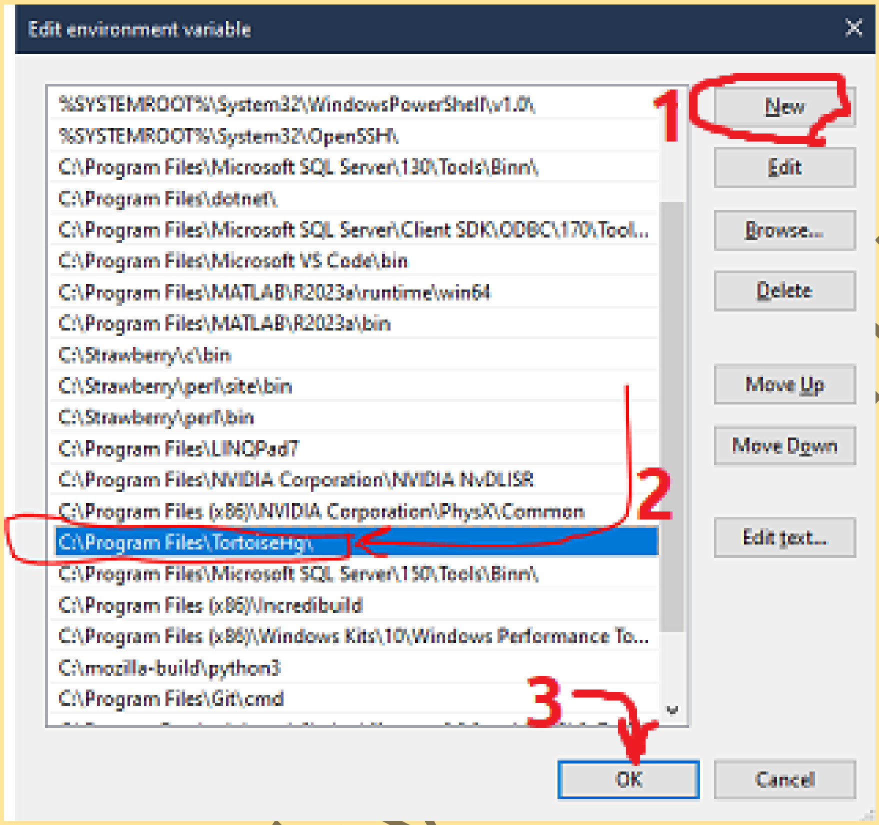
Variable	Value
OneDrive	C:\Users\pc\OneDrive
Path	C:\Users\pc\.cargo\bin;C:\Users\pc\AppData\Local\Programs\Python\Python312\Scripts\;C:\Users\pc\AppData...
PyCharm	C:\Program Files\JetBrains\PyCharm 2023.1.3\bin;
TEMP	C:\Users\pc\AppData\Local\Temp
TMP	C:\Users\pc\AppData\Local\Temp

Variable	Value
NUMBER_OF_PROCESSORS	8
OS	Windows_NT
Path	C:\Program Files (x86)\Common Files\Oracle\Java\javapath;C:\Windows\system32;C:\Windows;C:\Windows...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PHPSTORM_VM_OPTIONS	C:\Users\Public\jetbra\vmoptions\phpstorm.vmoptions
PROCESSOR_ARCHITECTURE	AMD64
PROCESSOR_IDENTIFIER	Intel64 Family 6 Model 142 Stepping 10, GenuineIntel
PROCESSOR_LEVEL	6
PROCESSOR_REVISION	8e0a

2.3. Edit Path variable:

Now copy path of Tortoise Hg and insert it as a new system environment variable as shown in the following screen shots.

- ❖ Note that the default path of Tortoise Hg varies among systems, but the most common path is ***C:\Program Files\TortoiseHg\***
  ⚠ To make sure about the path, search the ToroiseHg.exe file in your system's explorer. It is recommended to use Everything app, which you can download from https://www.voidtools.com/downloads/



```
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files\Microsoft SQL Server\130\Tools\Binn\
C:\Program Files\dotnet\
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\170\Tool...
C:\Program Files\Microsoft VS Code\bin
C:\Program Files\MATLAB\R2023a\runtime\win64
C:\Program Files\MATLAB\R2023a\bin
C:\Strawberry\c\bin
C:\Strawberry\perl\site\bin
C:\Strawberry\perl\bin
C:\Program Files\LINQPad7
C:\Program Files\NVIDIA Corporation\NVIDIA NvDLISR
C:\Program Files (x86)\NVIDIA Corporation\PhysX\Common
C:\Program Files\TortoiseHg\
C:\Program Files\Microsoft SQL Server\150\Tools\Binn\
C:\Program Files (x86)\Incredibuild
C:\Program Files (x86)\Windows Kits\10\Windows Performance To...
C:\mozilla-build\python3
C:\Program Files\Git\cmd
```

2.4. Test the Tortoise Hg configuration by running "hg" in CMD; and if you have configured it correctly; you will see "*Mercurial Distributed SCM*" message, and you are good to move on the next steps to create Mozilla Firefox installer.

1.3.3. Now that you have cloned Mozilla Build on your system, you should change your directory to where you have cloned Mozilla Build packages. Try typing the below command in **CMD** 👇

```bash
cd C:/Windows/System32/mozilla-central
```

❖ If it didn't work, just search for "mozilla-central" in *Everything* app , then copy and replace it's path after *cd*.

2nd Step: Build Environment Configuration

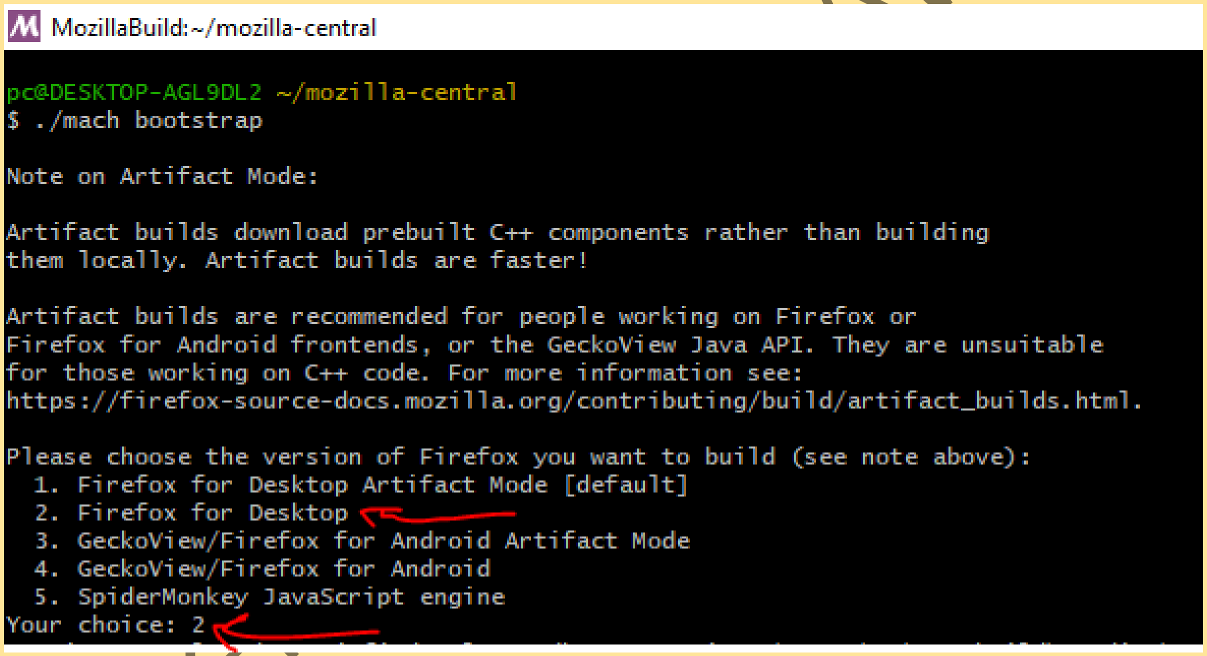
2.1. Download Mozilla-Build pack from https://www.dropbox.com/scl/fi/n5fd0a1118hquqhjt0blk/mozilla-build.zip?rlkey=iectp5uqhtwvh9f00x3r8xu6o&st=dbib6v27&dl=0

Password for extracting zip: **persianfoxproject.org**

2.2. In /mozilla-build directory, run .bat file, which is a shell command interface.

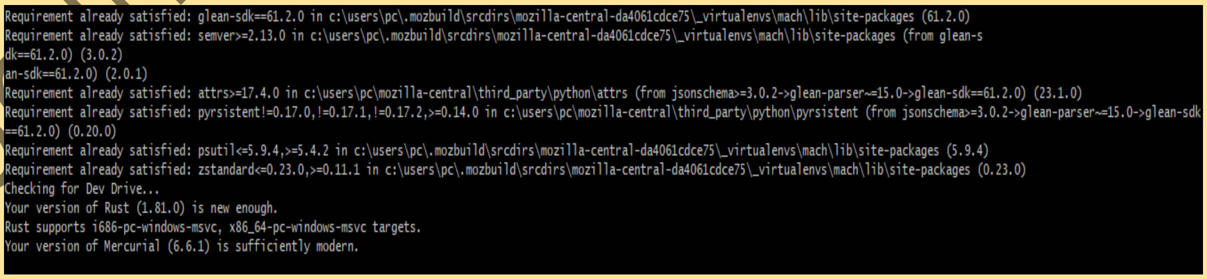
2.3. Change directory to path where you have cloned Mozilla Firefox source files in mozilla-central directory.

2.3. Run "**./mach bootstrap**" in the shell. Then you should see the following output from MozillaBuild CLI (Command Line Interface) 👇



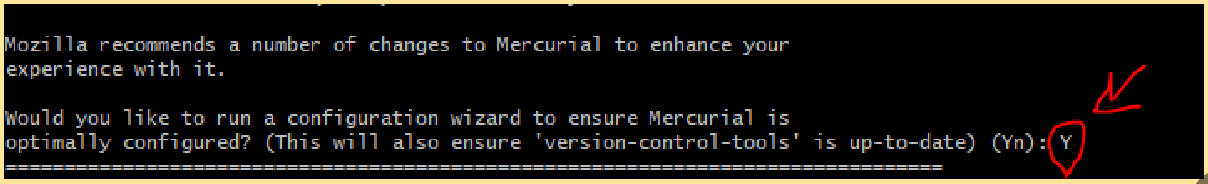
```
pc@DESKTOP-AGL9DL2 ~/mozilla-central
$ ./mach bootstrap

Note on Artifact Mode:

Artifact builds download prebuilt C++ components rather than building
them locally. Artifact builds are faster!

Artifact builds are recommended for people working on Firefox or
Firefox for Android frontends, or the GeckoView Java API. They are unsuitable
for those working on C++ code. For more information see:
https://firefox-source-docs.mozilla.org/contributing/build/artifact_builds.html.

Please choose the version of Firefox you want to build (see note above):
  1. Firefox for Desktop Artifact Mode [default]
  2. Firefox for Desktop
  3. GeckoView/Firefox for Android Artifact Mode
  4. GeckoView/Firefox for Android
  5. SpiderMonkey JavaScript engine
Your choice: 2
```

Figure 2.3.1: After seeing this choice list, please type "2" in order to carry on preparing bootstrap environment for building installer file named "mozmake.exe".



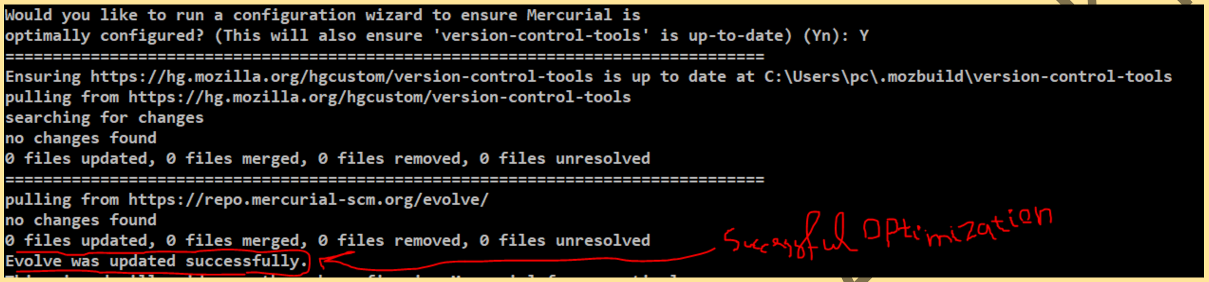
```
Requirement already satisfied: glean-sdk==61.2.0 in c:\users\pc\.mozbuild\srcdirs\mozilla-central-da4061cdce75\_virtualenvs\mach\lib\site-packages (61.2.0)
Requirement already satisfied: semver>=2.13.0 in c:\users\pc\.mozbuild\srcdirs\mozilla-central-da4061cdce75\_virtualenvs\mach\lib\site-packages (from glean-s
dk==61.2.0) (3.0.2)
an-sdk==61.2.0) (2.0.1)
Requirement already satisfied: attrs>=17.4.0 in c:\users\pc\mozilla-central\third_party\python\attrs (from jsonschema>=3.0.2->glean-parser~=15.0->glean-sdk==61.2.0) (23.1.0)
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.14.0 in c:\users\pc\mozilla-central\third_party\python\pyrsistent (from jsonschema>=3.0.2->glean-parser~=15.0->glean-sdk
==61.2.0) (0.20.0)
Requirement already satisfied: psutil<=5.9.4,>=5.4.2 in c:\users\pc\.mozbuild\srcdirs\mozilla-central-da4061cdce75\_virtualenvs\mach\lib\site-packages (5.9.4)
Requirement already satisfied: zstandard<=0.23.0,>=0.11.1 in c:\users\pc\.mozbuild\srcdirs\mozilla-central-da4061cdce75\_virtualenvs\mach\lib\site-packages (0.23.0)
Checking for Dev Drive...
Your version of Rust (1.81.0) is new enough.
Rust supports i686-pc-windows-msvc, x86_64-pc-windows-msvc targets.
Your version of Mercurial (6.6.1) is sufficiently modern.
```

Figure 2.3.2: Successful output of second choice, which is the correct one to build Firefox installer for desktop platforms, according to prompts and logs of MozillaBuild CLI.



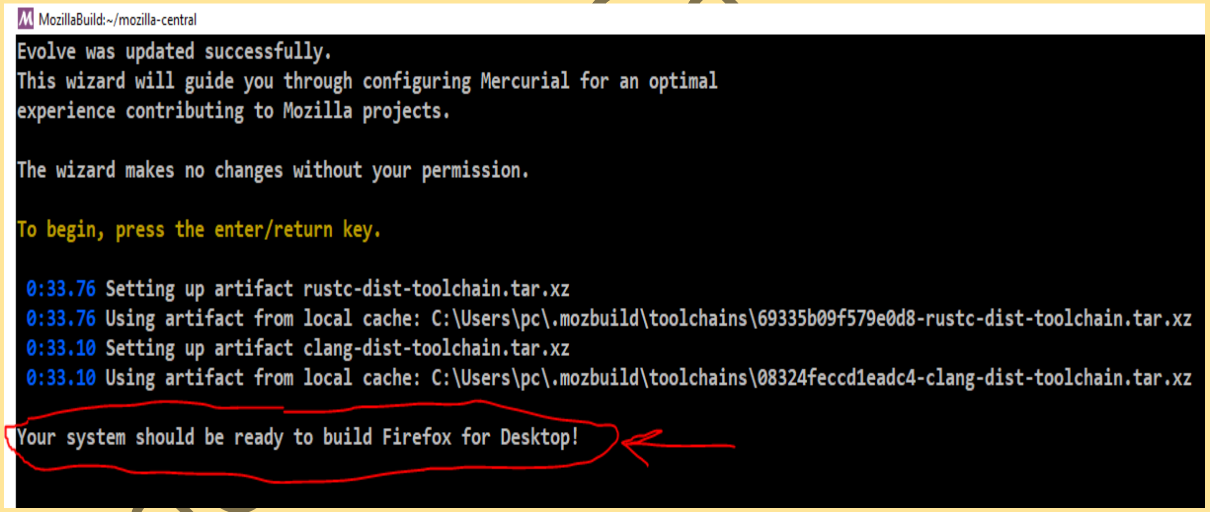
```
Mozilla recommends a number of changes to Mercurial to enhance your
experience with it.

Would you like to run a configuration wizard to ensure Mercurial is
optimally configured? (This will also ensure 'version-control-tools' is up-to-date) (Yn): Y
=================================================================================
```

Figure 2.3.3: Then CLI asks you whether you want to configure Mercurial optimally or not. It is better for you that you agree and type "Y", then hit "Enter" key to let MozillaBuild CLI configure Mercurial optimally.



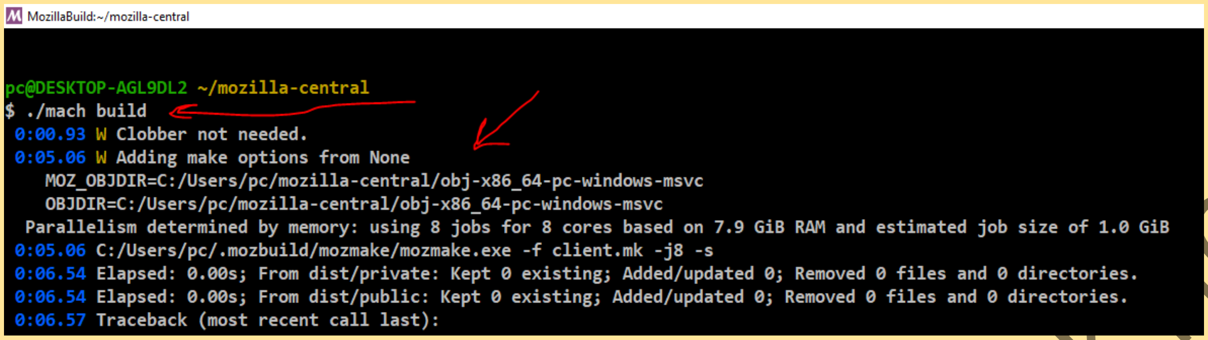
```
Would you like to run a configuration wizard to ensure Mercurial is
optimally configured? (This will also ensure 'version-control-tools' is up-to-date) (Yn): Y
================================================================================
Ensuring https://hg.mozilla.org/hgcustom/version-control-tools is up to date at C:\Users\pc\.mozbuild\version-control-tools
pulling from https://hg.mozilla.org/hgcustom/version-control-tools
searching for changes
no changes found
0 files updated, 0 files merged, 0 files removed, 0 files unresolved
================================================================================
pulling from https://repo.mercurial-scm.org/evolve/
no changes found
0 files updated, 0 files merged, 0 files removed, 0 files unresolved
Evolve was updated successfully.
```

Figure 2.3.4: To make sure about your Mercurial configuration's accuracy, please check the console log; so that if the operation has been successful, you can see the red-highlighted message from CLI's LOG.



```
Evolve was updated successfully.
This wizard will guide you through configuring Mercurial for an optimal
experience contributing to Mozilla projects.

The wizard makes no changes without your permission.

To begin, press the enter/return key.

 0:33.76 Setting up artifact rustc-dist-toolchain.tar.xz
 0:33.76 Using artifact from local cache: C:\Users\pc\.mozbuild\toolchains\69335b09f579e0d8-rustc-dist-toolchain.tar.xz
 0:33.10 Setting up artifact clang-dist-toolchain.tar.xz
 0:33.10 Using artifact from local cache: C:\Users\pc\.mozbuild\toolchains\08324feccd1eadc4-clang-dist-toolchain.tar.xz

Your system should be ready to build Firefox for Desktop!
```

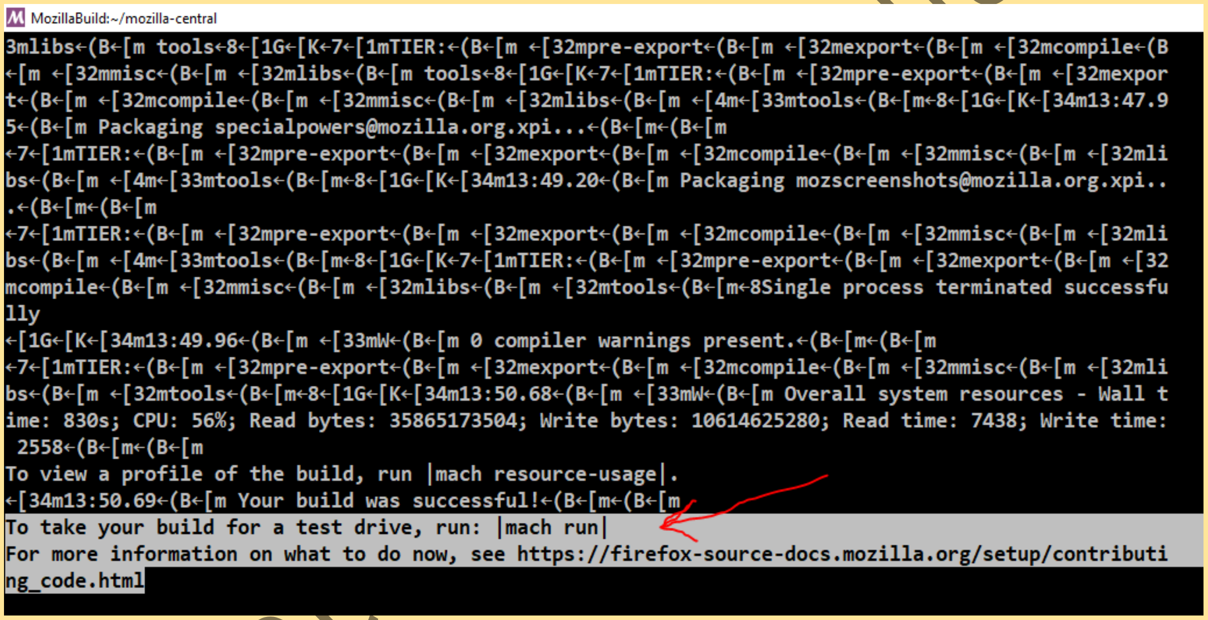
Figure 2.3.5: Now you only need to hit "*__Enter__*" key; so that the process would be good to go on until you see the red-highlighted LOG indicating that we are ready to execute *"./mach build*" which is a UNIX-based command afterwards.



MozillaBuild:~/mozilla-central

```
pc@DESKTOP-AGL9DL2 ~/mozilla-central
$ ./mach build
 0:00.93 W Clobber not needed.
 0:05.06 W Adding make options from None
    MOZ_OBJDIR=C:/Users/pc/mozilla-central/obj-x86_64-pc-windows-msvc
    OBJDIR=C:/Users/pc/mozilla-central/obj-x86_64-pc-windows-msvc
  Parallelism determined by memory: using 8 jobs for 8 cores based on 7.9 GiB RAM and estimated job size of 1.0 GiB
 0:05.06 C:/Users/pc/.mozbuild/mozmake/mozmake.exe -f client.mk -j8 -s
 0:06.54 Elapsed: 0.00s; From dist/private: Kept 0 existing; Added/updated 0; Removed 0 files and 0 directories.
 0:06.54 Elapsed: 0.00s; From dist/public: Kept 0 existing; Added/updated 0; Removed 0 files and 0 directories.
 0:06.57 Traceback (most recent call last):
```

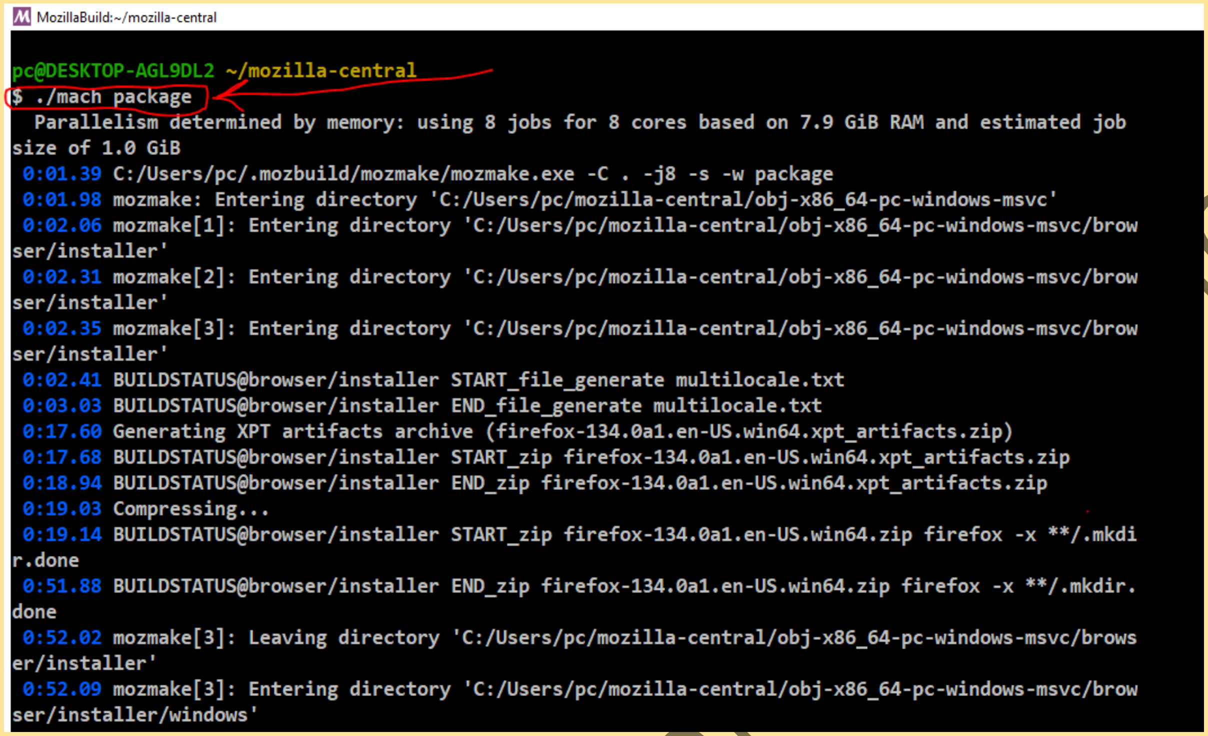
Figure 2.3.6: After successful bootstrapping, now you can build the Mozilla Firefox by running "*./mach build*" command in MozillaBuild CLI. Then result should look like the above LOG shot displaying RAM capacity and number of CPU cores. Otherwise you have had made a mistake before you reach this point.



MozillaBuild:~/mozilla-central

```
3mlibs←(B←[m tools←8←[1G←[K←7←[1mTIER:←(B←[m ←[32mpre-export←(B←[m ←[32mexport←(B←[m ←[32mcompile←(B
←[m ←[32mmisc←(B←[m ←[32mlibs←(B←[m tools←8←[1G←[K←7←[1mTIER:←(B←[m ←[32mpre-export←(B←[m ←[32mexpor
t←(B←[m ←[32mcompile←(B←[m ←[32mmisc←(B←[m ←[32mlibs←(B←[m ←[4m←[33mtools←(B←[m←8←[1G←[K←[34m13:47.9
5←(B←[m Packaging specialpowers@mozilla.org.xpi...←(B←[m←(B←[m
←7←[1mTIER:←(B←[m ←[32mpre-export←(B←[m ←[32mexport←(B←[m ←[32mcompile←(B←[m ←[32mmisc←(B←[m ←[32mli
bs←(B←[m ←[4m←[33mtools←(B←[m←8←[1G←[K←[34m13:49.20←(B←[m Packaging mozscreenshots@mozilla.org.xpi..
.←(B←[m←(B←[m
←7←[1mTIER:←(B←[m ←[32mpre-export←(B←[m ←[32mexport←(B←[m ←[32mcompile←(B←[m ←[32mmisc←(B←[m ←[32mli
bs←(B←[m ←[4m←[33mtools←(B←[m←8←[1G←[K←7←[1mTIER:←(B←[m ←[32mpre-export←(B←[m ←[32mexport←(B←[m ←[32
mcompile←(B←[m ←[32mmisc←(B←[m ←[32mlibs←(B←[m ←[32mtools←(B←[m←8Single process terminated successfu
lly
←[1G←[K←[34m13:49.96←(B←[m ←[33mW←(B←[m 0 compiler warnings present.←(B←[m←(B←[m
←7←[1mTIER:←(B←[m ←[32mpre-export←(B←[m ←[32mexport←(B←[m ←[32mcompile←(B←[m ←[32mmisc←(B←[m ←[32mli
bs←(B←[m ←[32mtools←(B←[m←8←[1G←[K←[34m13:50.68←(B←[m ←[33mW←(B←[m Overall system resources - Wall t
ime: 830s; CPU: 56%; Read bytes: 35865173504; Write bytes: 10614625280; Read time: 7438; Write time:
 2558←(B←[m←(B←[m
To view a profile of the build, run |mach resource-usage|.
←[34m13:50.69←(B←[m Your build was successful!←(B←[m←(B←[m
To take your build for a test drive, run: |mach run|
For more information on what to do now, see https://firefox-source-docs.mozilla.org/setup/contributi
ng_code.html
```

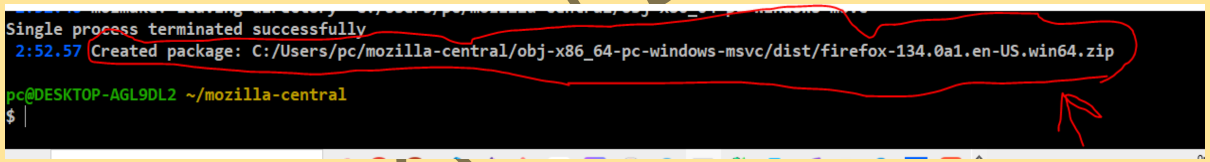
Figure 2.3.7: If your building process has been successfully executed, you should see the above highlighted message telling us about another command, which can be run to test our build process; afterwards you can package the installer executable file to setup your own modified Mozilla Firefox, using original or modified source code files.



MozillaBuild:~/mozilla-central

```
pc@DESKTOP-AGL9DL2 ~/mozilla-central
$ ./mach package
 Parallelism determined by memory: using 8 jobs for 8 cores based on 7.9 GiB RAM and estimated job
size of 1.0 GiB
 0:01.39 C:/Users/pc/.mozbuild/mozmake/mozmake.exe -C . -j8 -s -w package
 0:01.98 mozmake: Entering directory 'C:/Users/pc/mozilla-central/obj-x86_64-pc-windows-msvc'
 0:02.06 mozmake[1]: Entering directory 'C:/Users/pc/mozilla-central/obj-x86_64-pc-windows-msvc/brow
ser/installer'
 0:02.31 mozmake[2]: Entering directory 'C:/Users/pc/mozilla-central/obj-x86_64-pc-windows-msvc/brow
ser/installer'
 0:02.35 mozmake[3]: Entering directory 'C:/Users/pc/mozilla-central/obj-x86_64-pc-windows-msvc/brow
ser/installer'
 0:02.41 BUILDSTATUS@browser/installer START_file_generate multilocale.txt
 0:03.03 BUILDSTATUS@browser/installer END_file_generate multilocale.txt
 0:17.60 Generating XPT artifacts archive (firefox-134.0a1.en-US.win64.xpt_artifacts.zip)
 0:17.68 BUILDSTATUS@browser/installer START_zip firefox-134.0a1.en-US.win64.xpt_artifacts.zip
 0:18.94 BUILDSTATUS@browser/installer END_zip firefox-134.0a1.en-US.win64.xpt_artifacts.zip
 0:19.03 Compressing...
 0:19.14 BUILDSTATUS@browser/installer START_zip firefox-134.0a1.en-US.win64.zip firefox -x **/.mkdi
r.done
 0:51.88 BUILDSTATUS@browser/installer END_zip firefox-134.0a1.en-US.win64.zip firefox -x **/.mkdir.
done
 0:52.02 mozmake[3]: Leaving directory 'C:/Users/pc/mozilla-central/obj-x86_64-pc-windows-msvc/brows
er/installer'
 0:52.09 mozmake[3]: Entering directory 'C:/Users/pc/mozilla-central/obj-x86_64-pc-windows-msvc/brow
ser/installer/windows'
```

Figure 2.3.8: After you see the previous guidance message providing a URL of user-manual article, now we should create a package of setup files for our customized Mozilla Firefox.



```
Single process terminated successfully
 2:52.57 Created package: C:/Users/pc/mozilla-central/obj-x86_64-pc-windows-msvc/dist/firefox-134.0a1.en-US.win64.zip

pc@DESKTOP-AGL9DL2 ~/mozilla-central
$ |
```

Figure 2.3.9: Finally your installer executable file is ready.

R&D, Documentation, Deployment and Installation

By

SanaAllah Kheiri

Autumn semester 2024