

이제는 Web도 쉽게 Qt를 이용해 개발하자!

Qt for WebAssembly

Second Edition, Ver 2.0



Qt 한국 개발자 커뮤니티

www.qt-dev.com

예수님은 당신을 사랑합니다.

Qt for WebAssembly

Release version 2.0 (2024.07.21)

Homepage URL www.qt-dev.com

머리말

Qt WebAssembly에 관심 있는 분들께 조금이나마 도움이 되는 마음으로 문서 파일 그대로, 무료로 독자 분들께 배포합니다.

한가지 바라는 게 있다면, 아직도 믿음이 없는 분들이 복음을 듣고 하나님 아버지께 돌아오길 기도합니다. 또한 독생자 이신 예수님이 이 땅에 오셔서 우리의 죄를 대신 짊어지셨습니다. 존귀하신 예수님이 보혈로 우리의 죄를 사하여 주셨습니다. 하나님 아버지는 자식을 사랑하는 마음으로 믿지 않는 모든 이들이 하나님 아버지께 돌아오길 기다리고 계십니다.

이 책을 접하신 분들 중 아직도 믿음이 없는 분들이 계시다면 예수님을 구주로 영접하고 믿음의 자녀로 거듭나길 간절한 마음으로 기도하고 축원합니다. 또한 하나님의 선하신 은혜가 여러분과 함께하길 기도합니다. 아멘.

37. 예수께서 그에게 말씀하셨다. '네 마음을 다하고, 네 목숨을 다 하고, 네 뜻을 다하여, 주 너의 하나님을 사랑하여라' 하였으니, 38. 이것이 가장 중요하고 으뜸 가는 계명이다. 39. 둘째 계명도 이것과 같은데, '네 이웃을 네 몸과 같이 사랑하여라' 한 것이다.

[새번역성경] 마태복음 22:37~39

Table of Contents

1. What is Qt for WebAssembly	1
2. MS Windows에서 개발환경 구축	5
3. Linux에서 개발환경 구축	32
4. macOS에서 개발환경 구축	47
5. Qt for WebAssembly 프로그래밍의 시작	58
6. Signal and Slot	67
7. Widget and Layouts	77
8. 기본 데이터 타입과 유용한 타입들	92
9. Qt에서 제공하는 유용한 Template class	106
10. QPainter 클래스를 이용한 2D Graphics	113
11. 크로마키 영상 처리 구현	127
12. Timer	133
13. Thread Programming	138

1. What is Qt for WebAssembly

우리가 Web 기반의 어플리케이션을 하기 위해서는 PHP, ASP, ASP.NET, JavaScript 등의 Web Script Language를 사용해야 한다.



하지만 WebAssembly의 기술을 이용하면 C++을 이용해 작성한 어플리케이션을 Web Browser에서 동작할 수 있도록 가능하게 해준다.

WebAssembly는 C++ 이외에도 C, RUST 등 다양한 언어로 작성한 소스코드를 Byte 코드로 변환한 다음 Web Browser에서 실행이 가능하다.

또한 Qt Framework은 WebAssembly 기반의 어플리케이션을 쉽게 개발할 수 있는 방법을 제공한다. 이는 개발자에게 개발 시간을 단축할 수 있는 큰 장점을 가지고 있다.

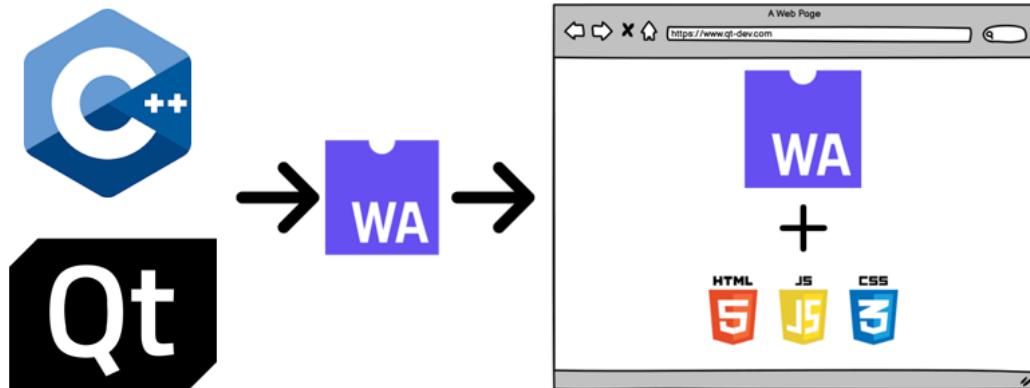


예를 들어 C++을 이용해 WebAssembly 기반의 웹 어플리케이션을 개발 시 처음부터 모든 필요한 라이브러리를 구현하는 개발자는 없을 것이다.

Qt Framework를 이용하면, C++을 이용해 WebAssembly 기반의 웹 어플리케이션을 개발 시, 방대한 분야의 라이브러리를 제공하기 때문에 개발 시간을 상당히 단축할 수 있

예수님은 당신을 사랑합니다.

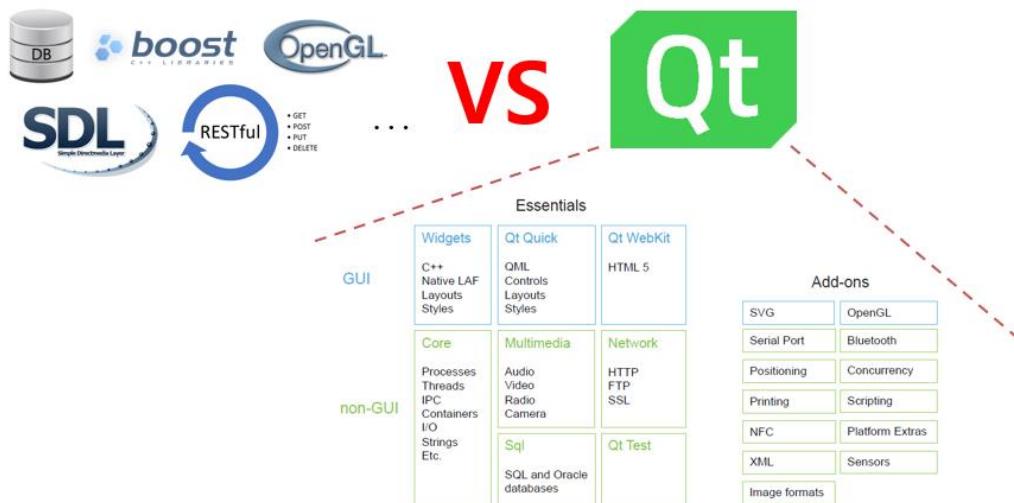
다. 그리고 디버깅도 Qt Creator 가능하기 때문에 매우 편리하게 디버깅 기술을 이용할 수 있다.



따라서 WebAssembly 기반의 어플리케이션 개발 시 Qt Framework을 이용하는 것은 개발자에게 가장 좋은 방법이 될 것이다.

만약 Qt Framework를 사용하지 않는 경우를 가정해 보자. 그렇다면 다양한 필요한 라이브러리를 직접 포팅 해야 한다. 예를 들어 DB, OpenGL, GUI 관련 등 다양한 라이브러리를 직접 포팅 해서 사용해야 한다. 하지만 이러한 포팅 작업은 매우 힘들다. 왜냐하면 포팅 시, 문제가 발생하면 직접 해결해야 하기 때문이다.

하지만 Qt Framework에서 제공하는 Qt for WebAssembly를 사용하면 Qt에서 제공하는 방대한 라이브러리를 사용할 수 있다는 장점이 있다. 따라서 Qt Framework를 사용하는 것은 가장 좋은 선택이 될 수 있다.



더불어 WebAssembly에서 가장 큰 걸림돌 중에 Multi Thread를 사용할 수 있느냐가 매

예수님은 당신을 사랑합니다.

우 중요하다.

WebAssembly 기술을 이용해 어플리케이션 구현 시, Thread 구조가 Single Thread 구조 이므로 Multi Thread를 사용하는 것이 매우 어렵다. Multi Thread를 사용해야 하는 어플리케이션 구현 시, 직접 이 문제를 해결해야 한다.

하지만 Qt Framework에서는 이미 Multi Thread를 사용이 가능하다는 장점이 있다.

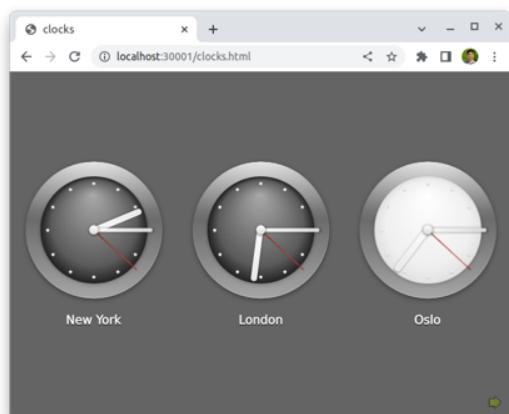
따라서 WebAssembly 기반의 Web Application 개발 시, Qt For WebAssembly 기술을 이용하면 C++과 Qt를 이용해 Web Browser에서 동작하는 Web Application을 구현할 수 있다.

- Qt for WebAssembly를 이용해 빌드한 어플리케이션과 비교



C++/Qt를 이용해 컴파일러로 빌드한
어플리케이션을 실행한 화면

VS



C++/Qt for WebAssembly를 이용해 빌드한
실행파일을 웹 브라우저에서 실행한 화면

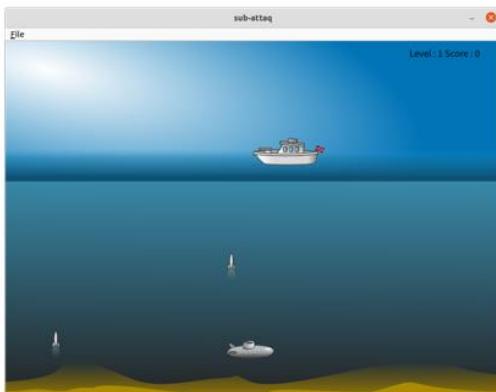
왼쪽의 실행 화면은 Desktop에서 빌드해 실행한 화면이고 오른쪽은 Web Browser에서 실행한 화면이다. 실행 환경은 다르지만 왼쪽과 오른쪽 모두 동일한

C++ 소스코드와 Qt 라이브러리를 사용했다는 것이다. 만약 Web Browser에서 동작하는 어플리케이션을 동일하게 만들어야 하는 경우 C++/Qt를 사용하지 않는다면 다른 웹 개발 언어를 사용해야 한다.

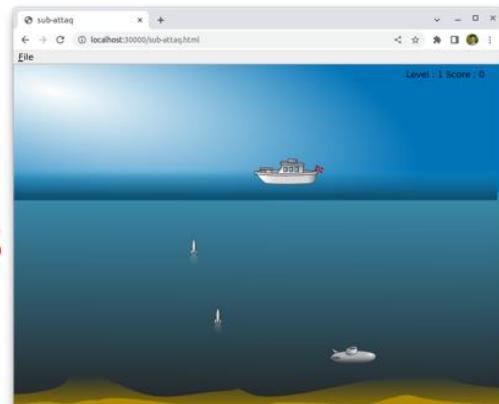
하지만 C++/Qt for WebAssembly를 사용하면 Web Browser에서 동작하는 어플리케이션도 C++과 Qt for WebAssembly를 사용해 구현할 수 있다.

따라서 C++ 과 Qt for WebAssembly를 이용하면 다른 언어를 사용하지 않고도 Web Browser에서 동작하는 어플리케이션과 Desktop에서 동작하는 어플리케이션이 동일한 소스코드로 구현이 가능하다는 장점이 있다.

예수님은 당신을 사랑합니다.



우분투 OS에서
어플리케이션 실행한 화면



크롬 웹 브라우저에서
웹 어플리케이션을 실행한 화면



MS Windows OS에서
어플리케이션 실행한 화면



네이버 웨일 웹 브라우저에서
웹 어플리케이션을 실행한 화면

미자막으로 여러분이 PHP, ASP, JSP, Ruby 등을 이용하지 않고 데스크탑에서 어플리케이션 구현 시 사용하는 C++과 Qt for WebAssembly 를 이용하면 데스크탑 어플리케이션과 Web Browser에서 동작하는 어플리케이션을 동일한 C++소스코드를 이용해 구현이 가능하다.

2. MS Windows에서 개발환경 구축

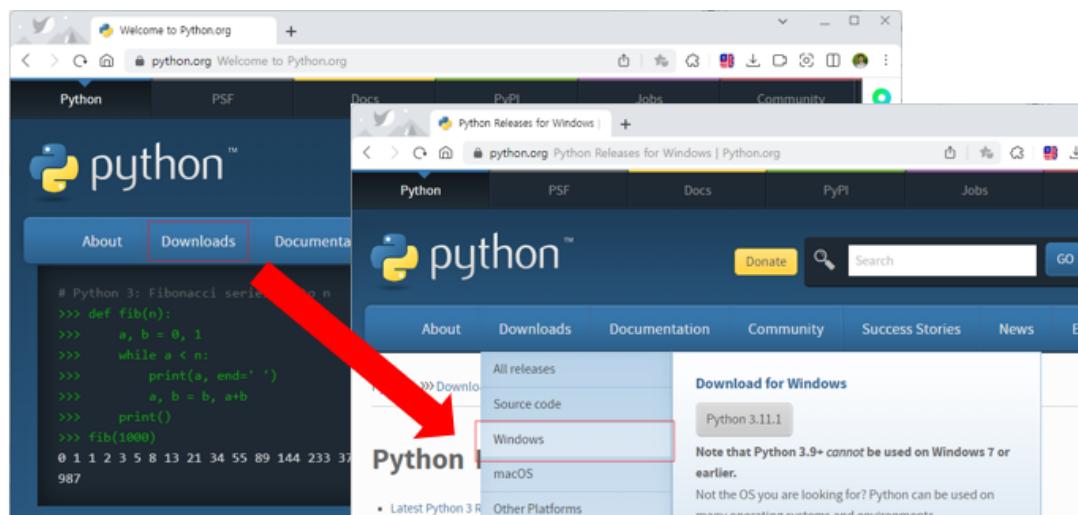
개발 환경을 구축하기 위해서 아래와 같은 항목을 설치해야 한다.

항목	설명
Python	간단한 웹서버 용으로 사용하기 위해서 설치
Git	Emscripten을 다운로드 받기 위한 목적으로 사용
Emscripten	WASM으로 빌드하기 위한 SDK 설치
Qt Framework	Qt버전은 Qt 6.5 LTS 버전을 사용

- Python 설치

Python 은 Qt WebAssembly를 사용하기 전에 직접 Emscripten SDK를 이용해 빌드한 후 Web Browser로 실행하기 위해서 웹서버가 필요하다. 따라서 Python 은 간단한 웹서버를 사용하기 위해서 필요하다. 추구 Qt 에서는 자동으로 웹서버를 로딩하기 때문에 Qt에서는 Python에서 제공하는 웹서버는 사용하지 않는다.

아래 그림에서 보는 것과 같이 python.org 사이트에 방문해 Python을 다운로드 받는다.



[Latest Python 3 Release - Python 3.11.1](#)

Stable Releases

- [Python 3.11.1 - Dec. 6, 2022](#)

Note that Python 3.11.1 cannot be used on Windows 7 or earlier.

- [Download Windows embeddable package \(32-bit\)](#)

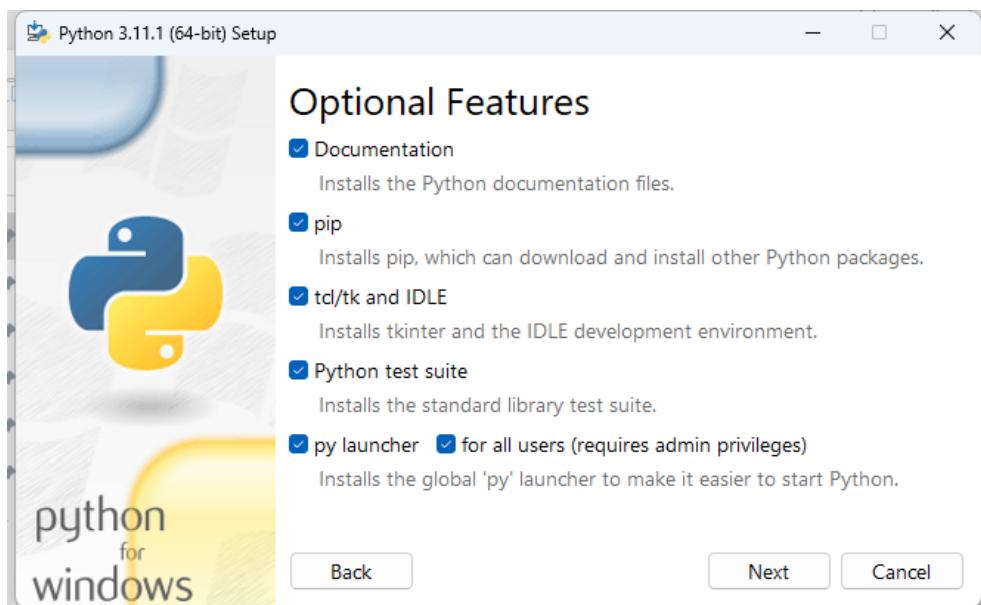
Pre-releases

- [Python 3.12.0a4 - Jan. 10, 2023](#)
- [Download Windows embeddable package \(32-bit\)](#)
- [Download Windows embeddable package \(64-bit\)](#)

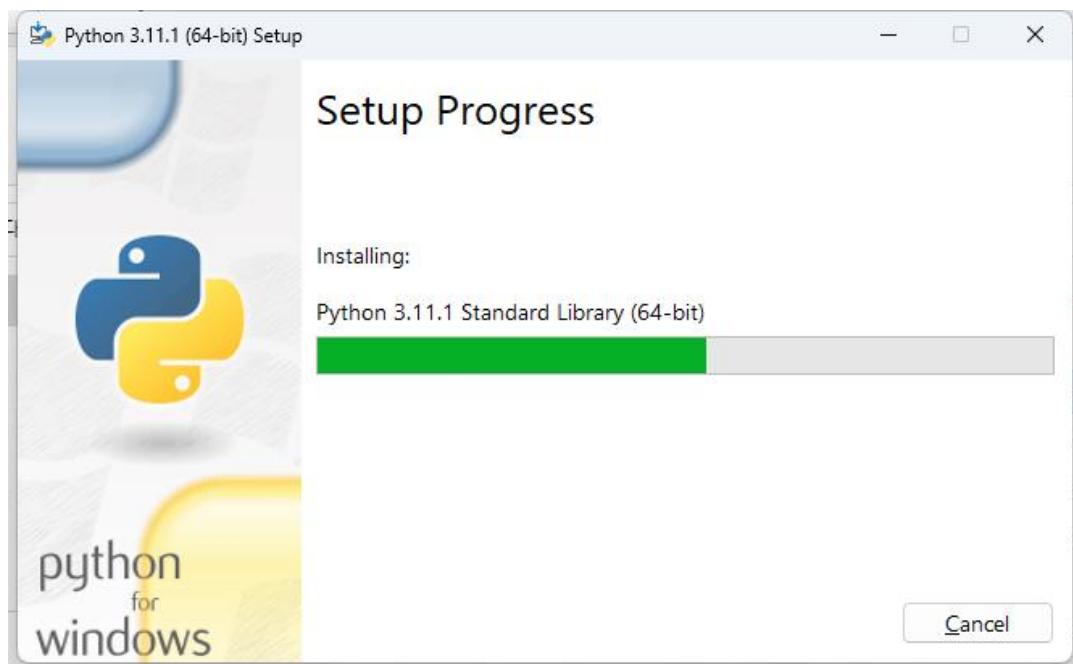
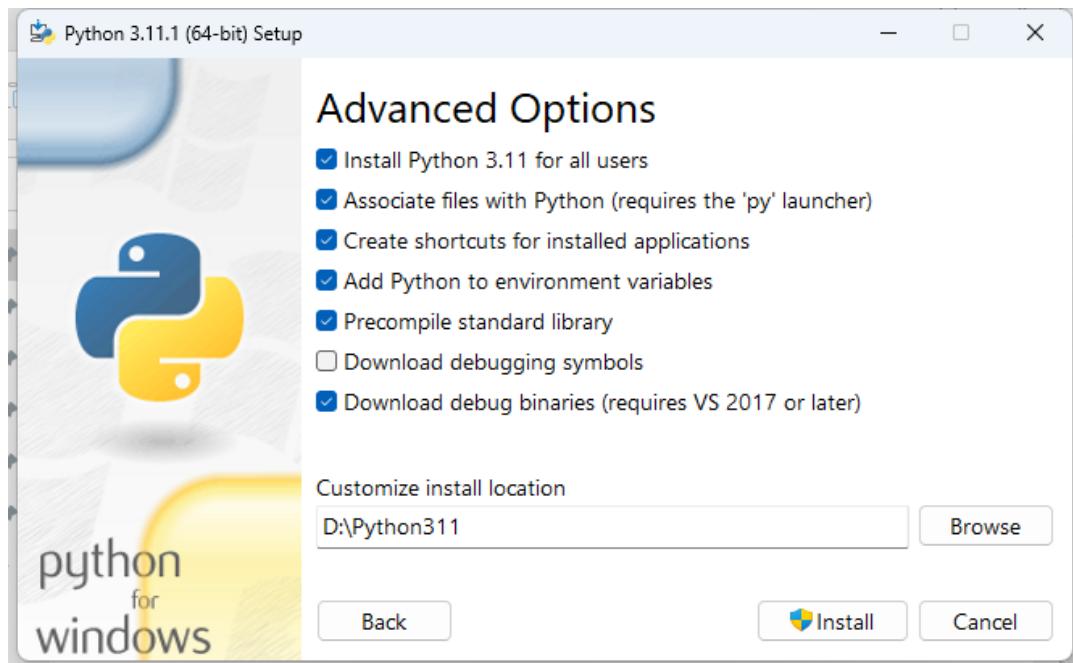
Version	Operating System	Description	MD5 Sum	File Size	GPG	Sigstore	
Gzipped source tarball	Source release		5c986b2865979b393aa50a31c65b64e8	26394378	SIG	CRT	SIG
XZ compressed source tarball	Source release		4efe92adf28875c77d3b9b2e8d3bc44a	19856648	SIG	CRT	SIG
macOS 64-bit universal2 installer	macOS	for macOS 10.9 and later	7c4d83ac21cf1e0470aa133ef6a1fff6	42665618	SIG	CRT	SIG
Windows embeddable package (32-bit)	Windows		cc960a3a6d5d1529117c463ac00aae43	9557137	SIG	CRT	SIG
Windows embeddable package (64-bit)	Windows		f16900451e15abe1ba3ea657f3c7fe9e	10538985	SIG	CRT	SIG
Windows embeddable package (ARM64)	Windows		405185d5ef1f436f8dbc370a868a2a85	9763968	SIG	CRT	SIG
Windows installer (32-bit)	Windows		a592f5db4f45ddc3a46c0ae465d3bee0	24054000	SIG	CRT	SIG
Windows installer (64-bit)	Windows	Recommended	3a02deed11f7ff4dbc1188d201ad164a	25218984	SIG	CRT	SIG
Windows installer (ARM64)	Windows	Experimental	3a98e0f9754199d99a7a97a6dacb0d91	24355528	SIG	CRT	SIG

다운로드 완료 후 아래에 보는 것과 같이 설치 파일을 실행한다. 다이얼로그 하단에 체크박스를 모두 체크한 후 [Customize installation] 링크 버튼을 클릭한다.

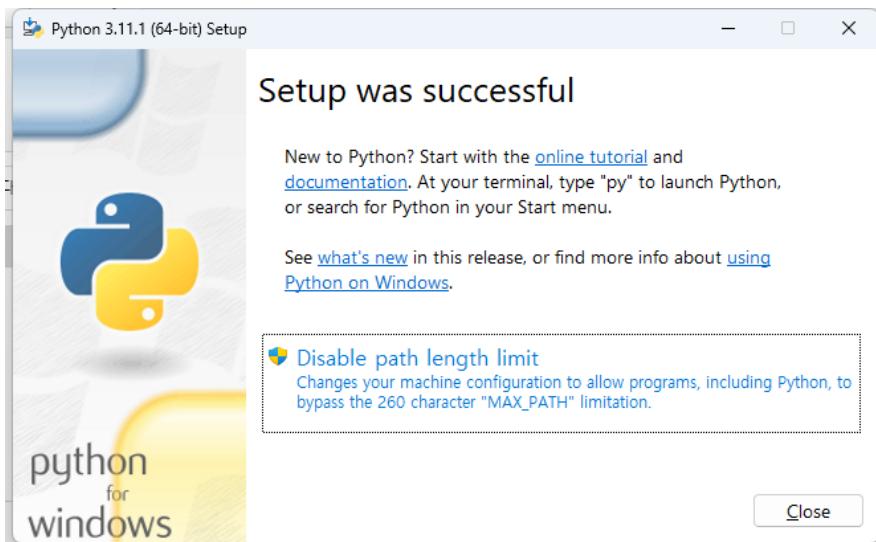
예수님은 당신을 사랑합니다.



예수님은 당신을 사랑합니다.



예수님은 당신을 사랑합니다.



● Git 설치

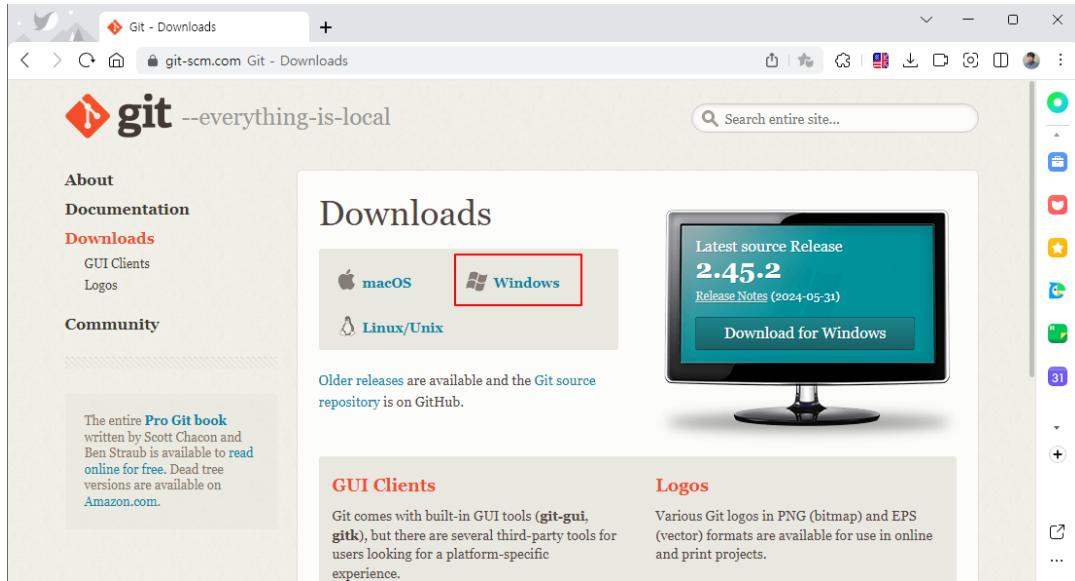
Git은 Emscripten을 다운로드 받기 위해서 사용한다. 하지만 Git을 사용하지 않더라도 Emscripten을 직접다운로드가 가능하다. 따라서 Git 설치는 필수가 아니다.

Git을 설치하기 위해서는 git-scm.org 사이트에서 아래와 같이 [Downloads] 링크를 클릭한다.

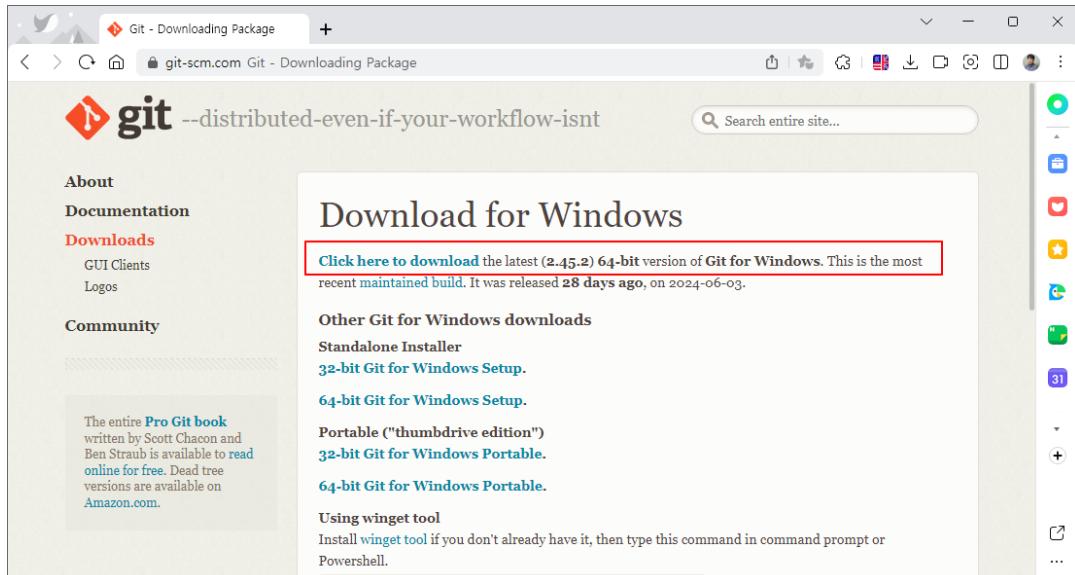
A screenshot of the git-scm.com website. The page features the Git logo and the tagline "--distributed-even-if-your-workflow-isn't". A search bar is at the top right. On the left, there are sections for "About", "Documentation", "Downloads" (which is highlighted with a red border), and "Community". On the right, there is a large image of a network of interconnected stacks of books, representing distributed version control. A monitor icon in the bottom right shows the latest source release version 2.45.2 and a "Download for Windows" button. A vertical sidebar on the far right contains icons for GitHub, Bitbucket, and other Git-related services.

예수님은 당신을 사랑합니다.

다음 페이지에서 [Windows] 링크를 클릭한다.

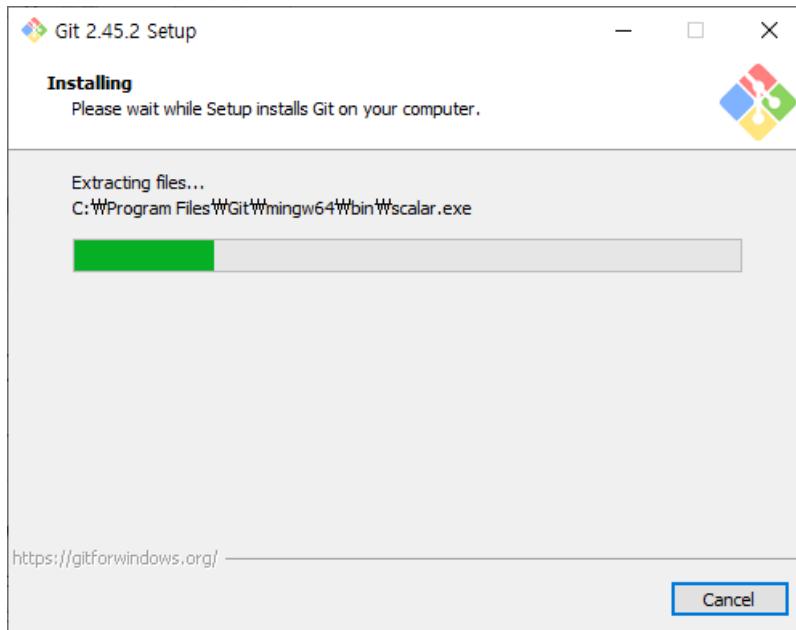


아래와 같이 [Click here to download] 링크를 클릭한다.



다운로드가 완료되면 아래 그림에서 보는 것과 같이 설치파일을 실행한다.

예수님은 당신을 사랑합니다.



● Emscripten 설치

Emscripten SDK는 WASM을 생성하기 위한 SDK이다. emscripten.org 사이트이다.

A screenshot of the emscripten.org documentation page. The URL in the address bar is "emscripten.org Main — Emscripten 3.1.61-git (dev) documentation". The page features a sidebar with links like "Introducing Emscripten", "Getting Started" (which is highlighted with a red box), "Compiling and Running Projects", etc. The main content area has a large "emscripten" logo with a lightning bolt icon. Below it, a text block states: "Emscripten is a complete compiler toolchain to WebAssembly, using LLVM, with a special focus on speed, size, and the Web platform." There are three columns: "Porting", "APIs", and "Fast". The "Porting" column describes how to compile existing projects. The "APIs" column explains how Emscripten converts OpenGL into WebGL. The "Fast" column discusses the performance benefits due to LLVM and WebAssembly.

위의 페이지에서 보는 것과 같이 [Getting Started] 링크를 클릭한다.

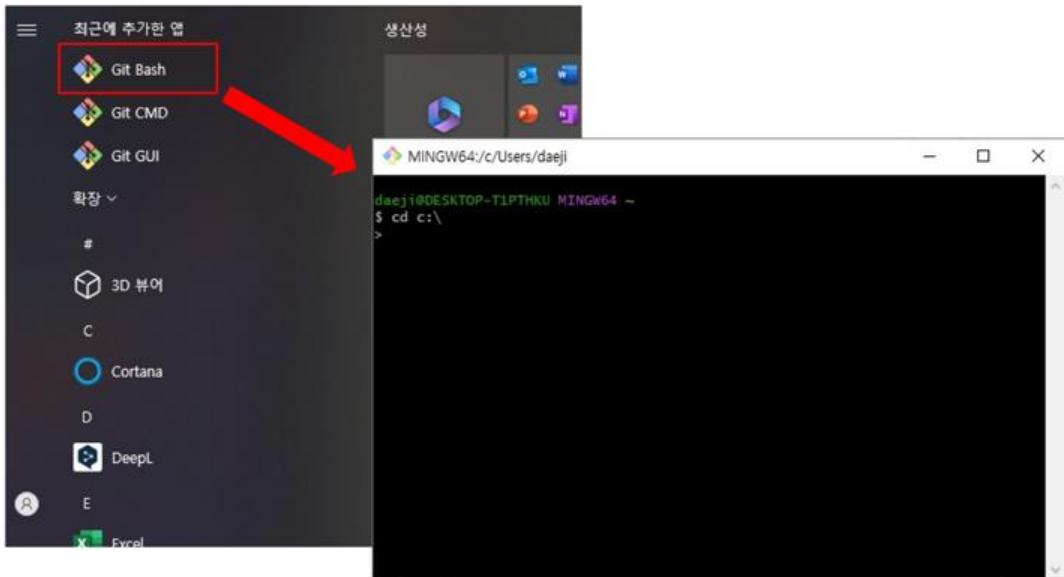
예수님은 당신을 사랑합니다.

The screenshot shows a web browser window displaying the Emscripten 'Getting Started' documentation. The URL is emscripten.org/Getting%20Started. The page has a blue header with the Emscripten logo and a search bar. The main content area has a dark background with white text. A sidebar on the left contains links for 'Introducing Emscripten', 'Getting Started' (which is currently selected), 'Compiling and Running Projects', and other sections like 'Porting' and 'API Reference'. The 'Getting Started' section includes links for 'Download and install', 'Emscripten Tutorial', 'Emscripten Test Suite', 'Bug Reporting', and 'FAQ'. A red box highlights the 'Download and install' link. The right side of the page has a 'Documentation', 'Downloads', and 'Community' navigation bar, and a 'Fork me on GitHub' button. A vertical toolbar on the right contains various icons for file operations.

다음으로 [Download and install] 링크를 클릭한다.

The screenshot shows a web browser window displaying the Emscripten 'Download and install' documentation. The URL is emscripten.org/Download%20and%20install. The page has a blue header with the Emscripten logo and a search bar. The main content area has a light blue background with white text. A sidebar on the left contains links for 'Introducing Emscripten', 'Getting Started' (which is currently selected), 'Compiling and Running Projects', and other sections like 'Porting' and 'API Reference'. The 'Getting Started' section includes links for 'Download and install', 'Emscripten Tutorial', 'Emscripten Test Suite', 'Bug Reporting', and 'FAQ'. The 'Download and install' link is highlighted with a red box. The right side of the page has a 'Documentation', 'Downloads', and 'Community' navigation bar, and a 'Fork me on GitHub' button. A vertical toolbar on the right contains various icons for file operations. The page content includes a 'Note' section with text about building from source and a 'Tip' section with text about using unofficial packages.

위의 페이지에서 보는 것과 같이 설치한 Git Bash 창을 실행한다.



그리고 아래와 같이 입력한다.

```
$ cd /d  
$ git clone https://github.com/emscripten-core/emsdk.git
```

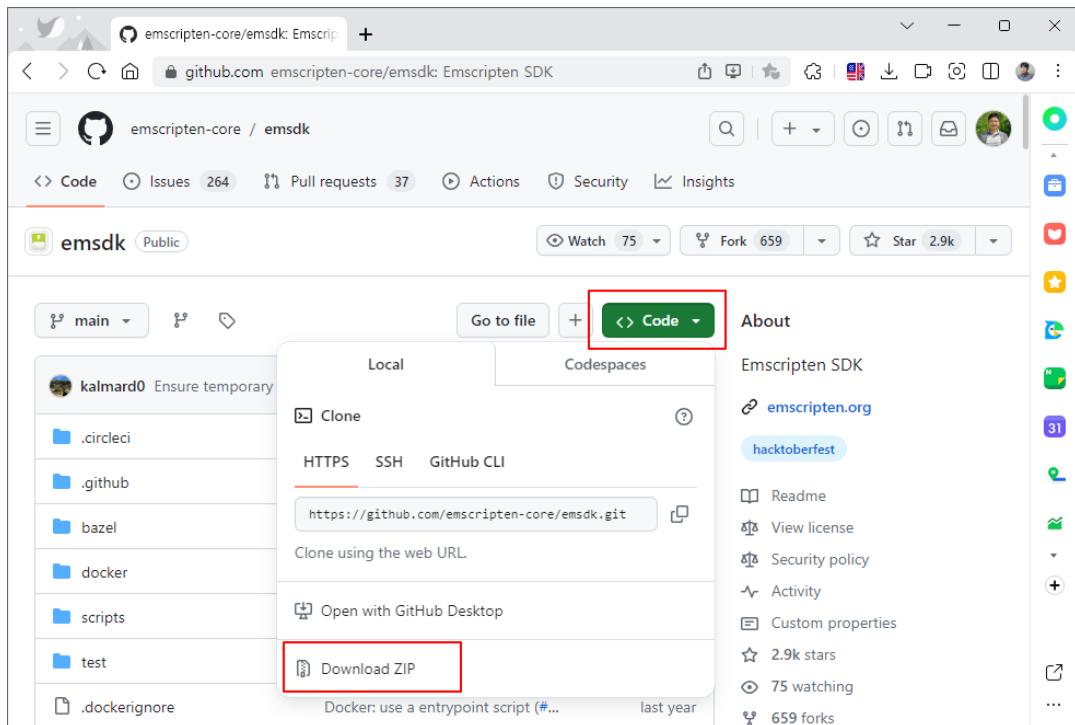
A screenshot of a terminal window titled 'MINGW64:/c'. The user has run the commands '\$ cd /d' and '\$ git clone https://github.com/emscripten-core/emsdk.git'. The output shows the cloning process: 'Cloning into 'emsdk'...', 'remote: Enumerating objects: 4083, done.', 'remote: Counting objects: 100% (21/21), done.', 'remote: Compressing objects: 100% (19/19), done.', 'remote: Total 4083 (delta 5), reused 12 (delta 1), pack-reused 4062.', 'Receiving objects: 100% (4083/4083), 2.23 MiB | 207.00 KiB/s, done.', 'Resolving deltas: 100% (2668/2668), done.'

위의 화면에서 보는 것과 같이 Git을 이용해 다운로드 받을 수 있다.

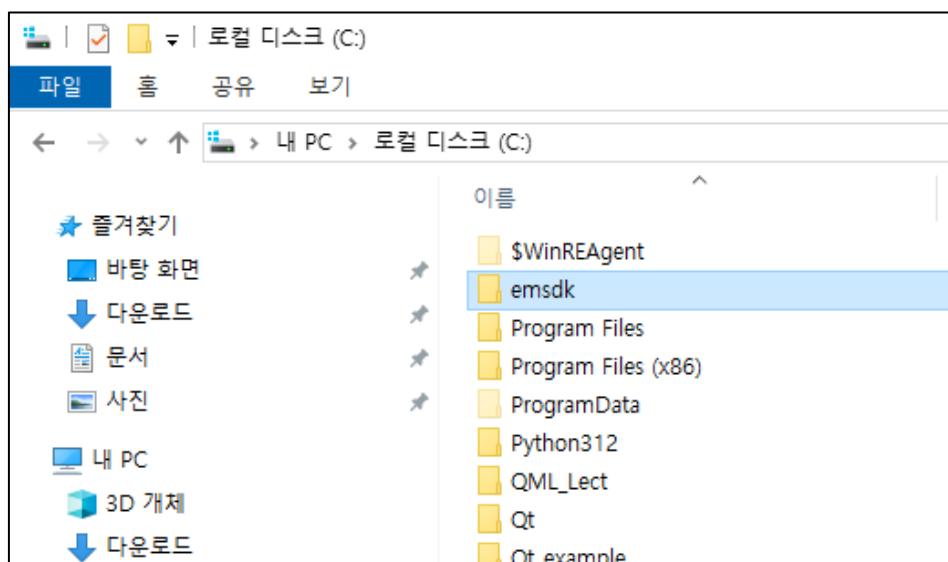
예수님은 당신을 사랑합니다.

<Git을 사용하지 않는 경우>

Web Browser에서 <https://github.com/emscripten-core/emsdk.git> 사이트를 방문한다. 그리고 아래와 같이 [Code] 링크를 클릭한 후, [Download ZIP] 링크를 클릭한다.

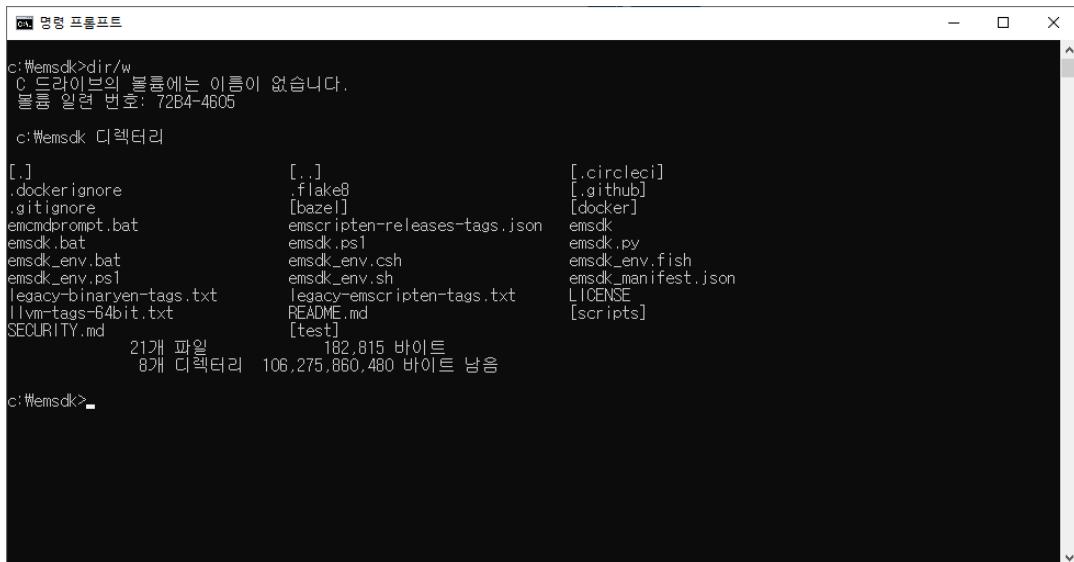


다운로드 받은 후, 압축해제 한 디렉토리 명을 emsdk 로 변경한다.



다음으로, 명령 프롬프트를 실행한다.

예수님은 당신을 사랑합니다.



```
c:\#emsdk>dir/w
C 드라이브의 폴더에는 이름이 없습니다.
폴더 일련 번호: 72B4-4605

c:\#emsdk 디렉터리

[.] [.] [.circleci]
.dockerignore .flake8 [.github]
.emscripten [.bazel] [.docker]
.emcmdprompt.bat emscripten-releases-tags.json emsdk
.emsdk.bat emsdk.ps1 emsdk.ps
.emsdk_env.bat emsdk_env.csh emsdk_env.sh
.emsdk_env.ps1 emsdk_env.sh legacy-emscripten-tags.txt
[legacy-binaryen-tags.txt] README.md [scripts]
[!vm-tags-64bit.txt] [test]
[SECURITY.md] 21개 파일 182,815 바이트
8개 디렉터리 106,275,860,480 바이트 남음

c:\#emsdk>
```

그리고 위의 그림에서 보는 것과 같이 emsdk 디렉토리로 이동한다. 다음으로 아래와 같이 Emscripten에서 최신버전을 설치한다.

```
D:\#emsdk> emsdk.bat install latest
...
Downloading: C:/emsdk/downloads/node-v18.20.3-win-x64.zip from
https://storage.googleapis.com/webassembly/emscripten-releases-builds/deps/node-
v18.20.3-win-x64.zip, 30476796 Bytes
Done installing tool 'node-18.20.3-64bit'.
Installing tool 'python-3.9.2-nuget-64bit'..
...
D:\#emsdk> emsdk.bat activate latest
Resolving SDK alias 'latest' to '3.1.61'
...
28e4a74b579b4157bda5fc34f23c7d3905a8bd6c-64bit'
Setting the following tools as active:
node-18.20.3-64bit
python-3.9.2-nuget-64bit
java-8.152-64bit
releases-28e4a74b579b4157bda5fc34f23c7d3905a8bd6c-64bit
...
```

예수님은 당신을 사랑합니다.

위와 같이 최종 버전을 설치하였다면 아래와 같이 최신 버전이 설치되어 있는지 확인 한다.

```
D:\emsdk> emsdk_env.bat
```

```
Setting up EMSDK environment (suppress these messages with EMSDK_QUIET=1)
```

```
Setting environment variables:
```

```
...
```

```
D:\emsdk> emcc --version
```

```
...
```

```
D:\emsdk> em++ --version
```

```
emcc (Emscripten gcc/clang-like replacement + linker emulating GNU ld) 3.1.61  
(67fa4c16496b157a7fc3377af69ee0445e8a6e3)
```

```
Copyright (C) 2014 the Emscripten authors (see AUTHORS.txt)
```

```
This is free and open source software under the MIT license.
```

```
There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR  
PURPOSE.
```

다음으로 Qt를 사용하기 전에 직접 간단한 "Hello World" 를 출력하는 소스코드를 작성 한 다음, 직접 Emscripten SDK를 이용해 빌드 후 실행해 보도록 하자. 아래에서 보는 것과 같이 hello.c 소스코드 파일을 작성한다.

```
#include <stdio.h>  
  
int main()  
{  
    printf("Hello World ~ ^~; \n");  
    return 1;  
}
```

위와 같이 작성한다음 아래와 같은 방법으로 빌드한다.

```
D:\Examples\00_hello_world_c_lang> dir
```

```
2024-07-03 오후 02:37 <DIR> .
```

```
2024-07-03 오후 02:37 <DIR> ..
```

예수님은 당신을 사랑합니다.

2024-07-01 오전 11:11

93 hello.c

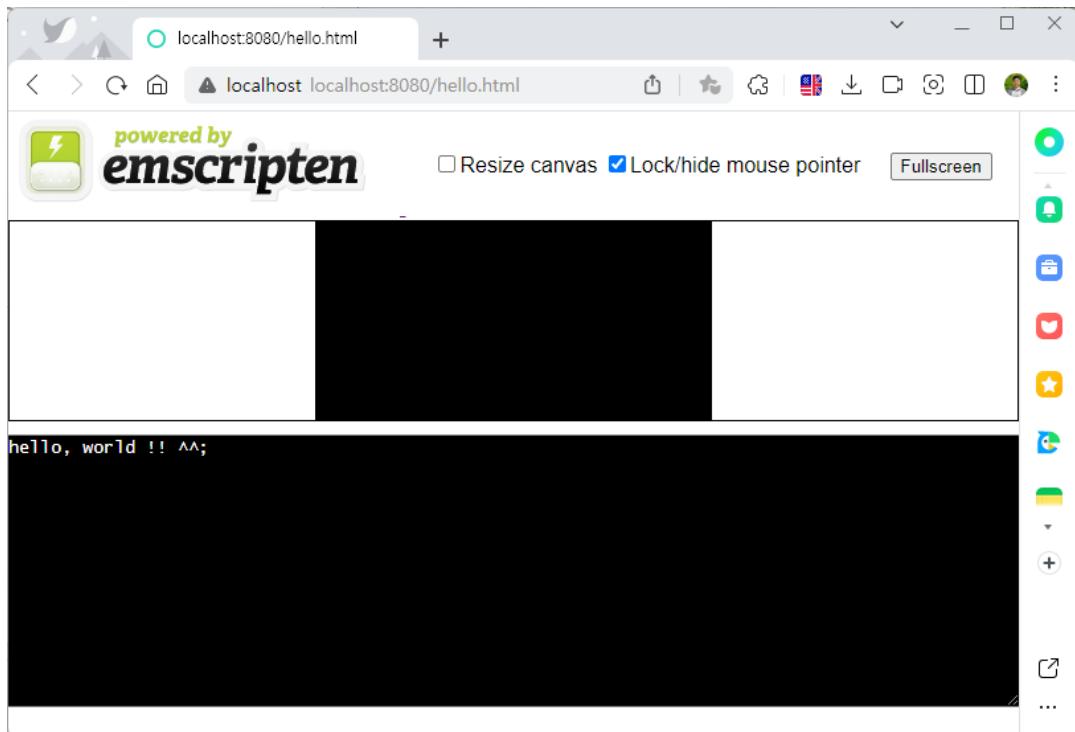
```
D:\Examples\00_hello_world_c_lang> emcc hello.c -s WASM=1 -o hello.html
```

```
D:\Examples\00_hello_world_c_lang> dir
```

```
2024-07-03 오후 02:37 <DIR> .
2024-07-03 오후 02:37 <DIR> ..
2024-07-01 오전 11:11 93 hello.c
2024-07-03 오후 02:37 103,800 hello.html
2024-07-03 오후 02:37 68,982 hello.js
2024-07-03 오후 02:37 12,091 hello.wasm
```

다음으로 동일한 디렉토리에서 Python에서 제공하는 웹서버를 로딩 한다.

```
D:\Examples\00_hello_world_c_lang> python -m http.server 8080
Serving HTTP on :: port 8080 (http://[::]:8080/) ...
```



위와 같이 Web Browser에서 Hello World가 출력된 것을 확인할 수 있다. 다음은 Qt Framework를 설치한다.

- Emscripten SDK에서 제공하는 웹서버 사용하기

Python에서 제공하는 웹서버 기능 이외에도 Emscripten SDK에서 제공하는 emrun 명령어는 웹서버를 통해서 HTML파일을 Web Browser에서 읽을 수 있다. 아래와 같이 사용하면 특정 디렉토리에 있는 HTML파일을 웹 서버를 통해서 실행할 수 있다.

```
emrun --browser chrome d:\root\index.html
```

--browser 뒤의 이름은 Web Browser의 이름이다. 지원 가능한 Web Browser 리스트는 아래와 같은 명령어를 통해 검색할 수 있다.

```
emrun -list_browsers
```

마지막 디렉토리 명과 파일명은 웹서버를 통해서 실행하고자 하는 HTML파일의 디렉토리 위치와 파일명이다. 위와 같이 실행하면 웹서버에 접근해서 접속하는 것과 같이 d:\root\index.html 파일을 Google Chrome Web Browser에서 실행해 준다.

- Qt Framework 설치

Qt 버전은 6.5.x LTS 버전을 설치한다. Qt 6.5.x 버전을 설치하기 위해서 아래와 같이 사이트에 방문한다.

Site 주소: <https://download.qt.io>

예수님은 당신을 사랑합니다.

Qt Downloads

Name	Last modified	Size	Metadata
snapshots/	22-Nov-2022 10:06	-	
online/	19-Nov-2020 14:24	-	
official_releases/	01-Dec-2022 09:12	-	
new_archive/	29-Jan-2021 15:13	-	
ministro/	20-Feb-2017 10:32	-	
linguist_releases/	26-Mar-2019 07:49	-	
learning/	24-Feb-2021 15:09	-	
development_releases/	06-May-2021 07:48	-	
community_releases/	23-Feb-2017 07:29	-	
archive/	10-Dec-2020 12:52	-	

다음으로 [online_installers] 링크를 클릭한다.

Index of /official_releases

Parent Directory	-
vsaddin/	05-Jan-2022 13:23
qtdesignstudio/	13-Jul-2022 20:21
qtcreator/	22-Nov-2022 09:56
qtchooser/	08-Oct-2018 07:53
qt3dstudio/	28-Oct-2020 14:22
qt/	29-Sep-2022 07:41
qt-installer-framework/	12-Dec-2022 09:50
qbs/	24-Nov-2022 11:48
pyside/	30-Nov-2015 13:39
online_installers/	12-Dec-2022 11:20
jom/	12-Dec-2018 15:13
gdb/	17-Nov-2014 13:42
additional_libraries/	03-Mar-2021 10:18

그리고 확장자가 exe 파일을 다운로드 받는다.

예수님은 당신을 사랑합니다.

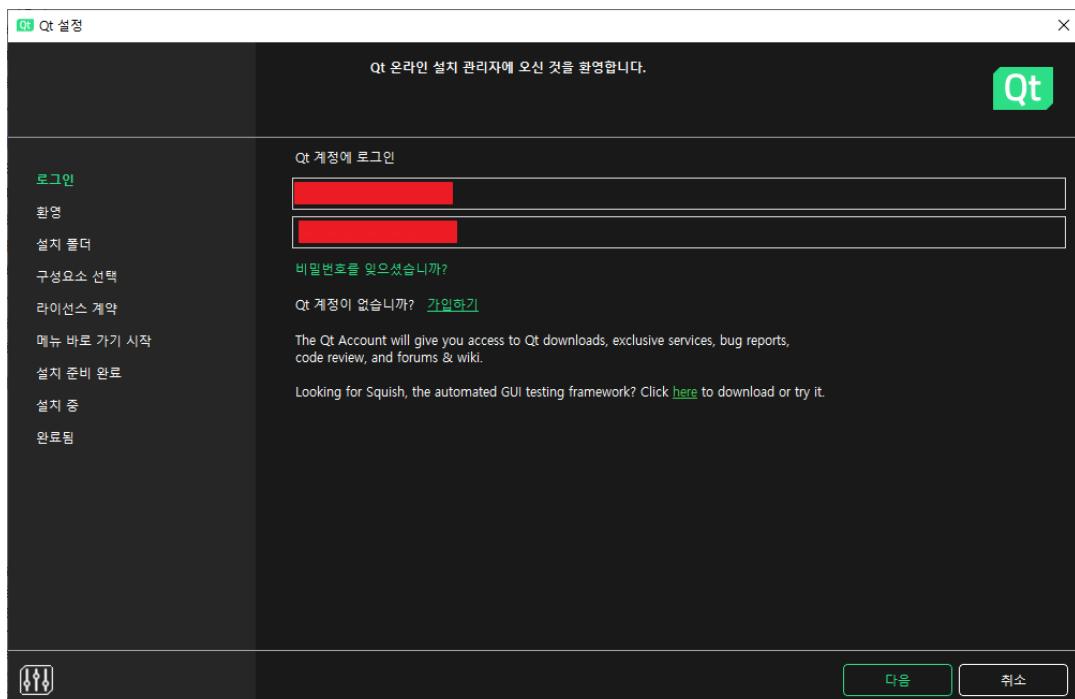
Qt Downloads

Name	Last modified	Size	Metadata
Parent Directory	-	-	-
qt-unified-windows-x64-online.exe	15-May-2024 10:07	47M	Details
qt-unified-mac-x64-online.dmg	15-May-2024 10:07	20M	Details
qt-unified-linux-x64-online.run	15-May-2024 10:07	66M	Details
qt-unified-linux-arm64-online.run	15-May-2024 10:07	68M	Details

For Qt Downloads, please visit [qt.io/download](#)

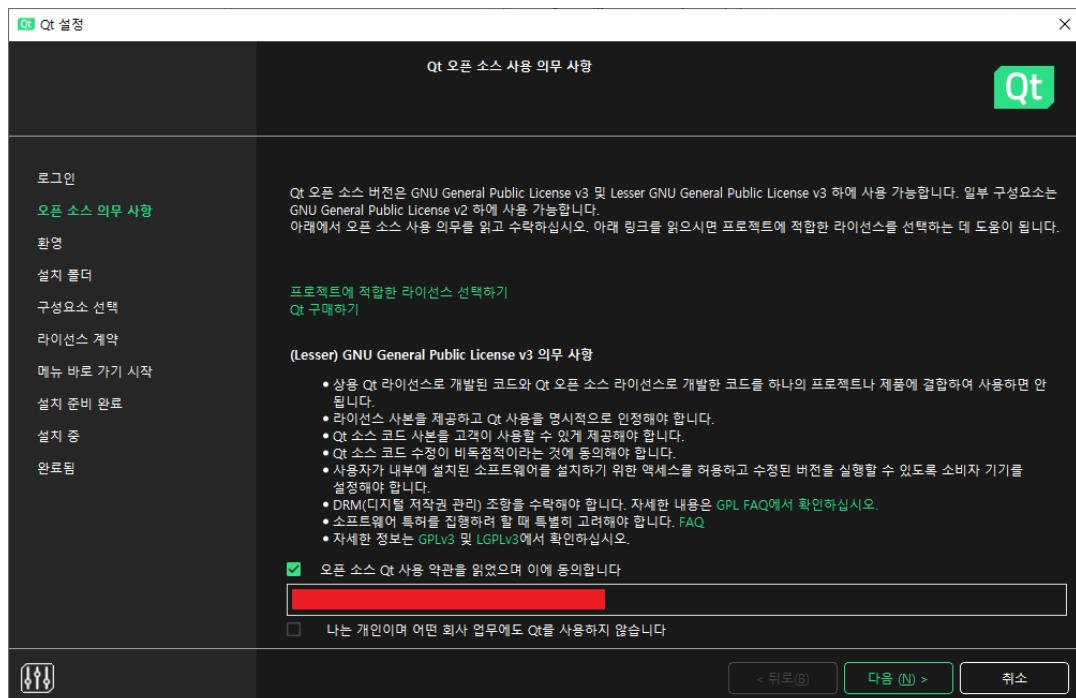
Qt® and the Qt logo is a registered trade mark of The Qt Company Ltd and is used pursuant to a license from The Qt Company Ltd.

다운로드 완료 후, 다운로드 받은 파일을 실행한다. Qt 계정 입력란에 계정을 입력하고 하단의 [다음] 버튼을 클릭한다.

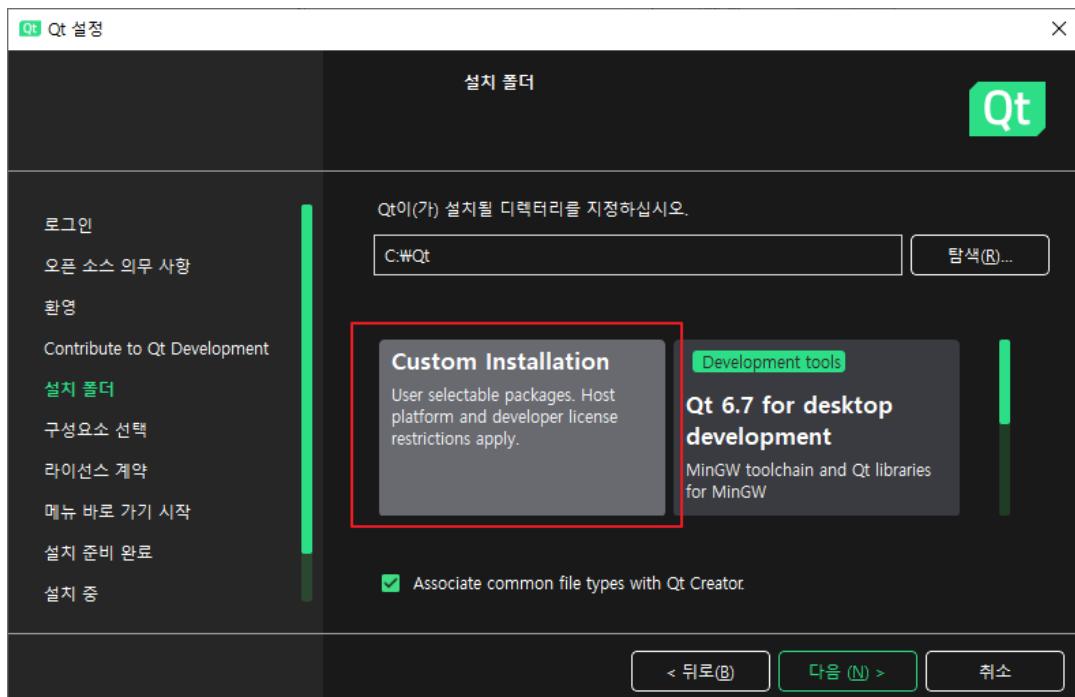
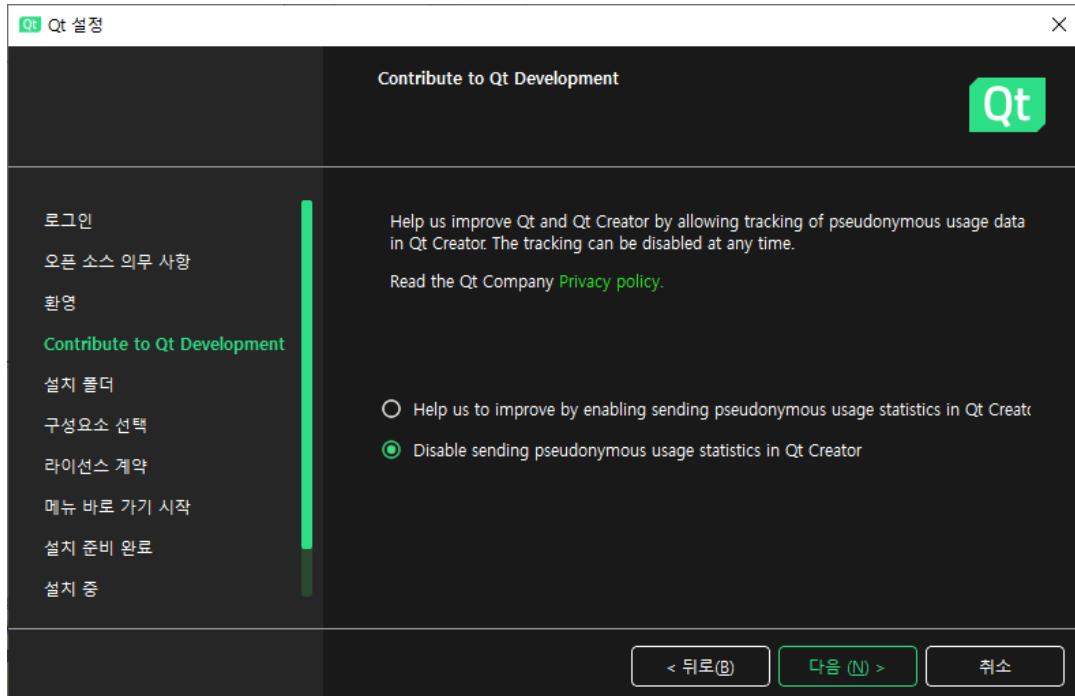


만약 계정이 없으면 위의 [가입하기] 링크버튼을 클릭한다.

예수님은 당신을 사랑합니다.



예수님은 당신을 사랑합니다.



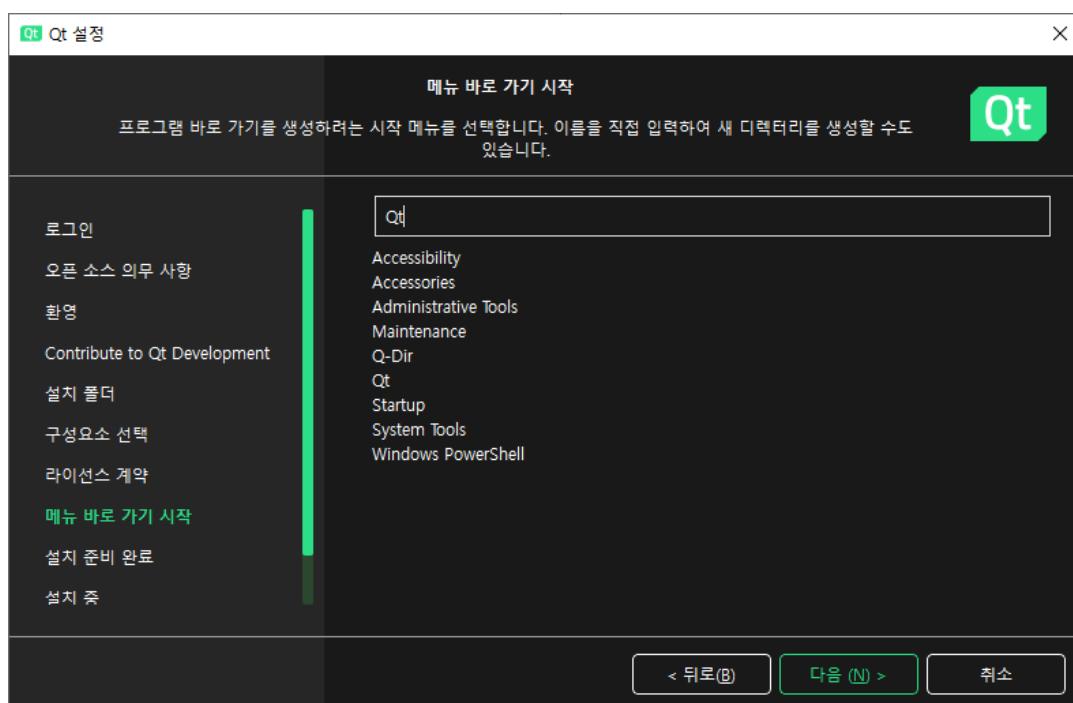
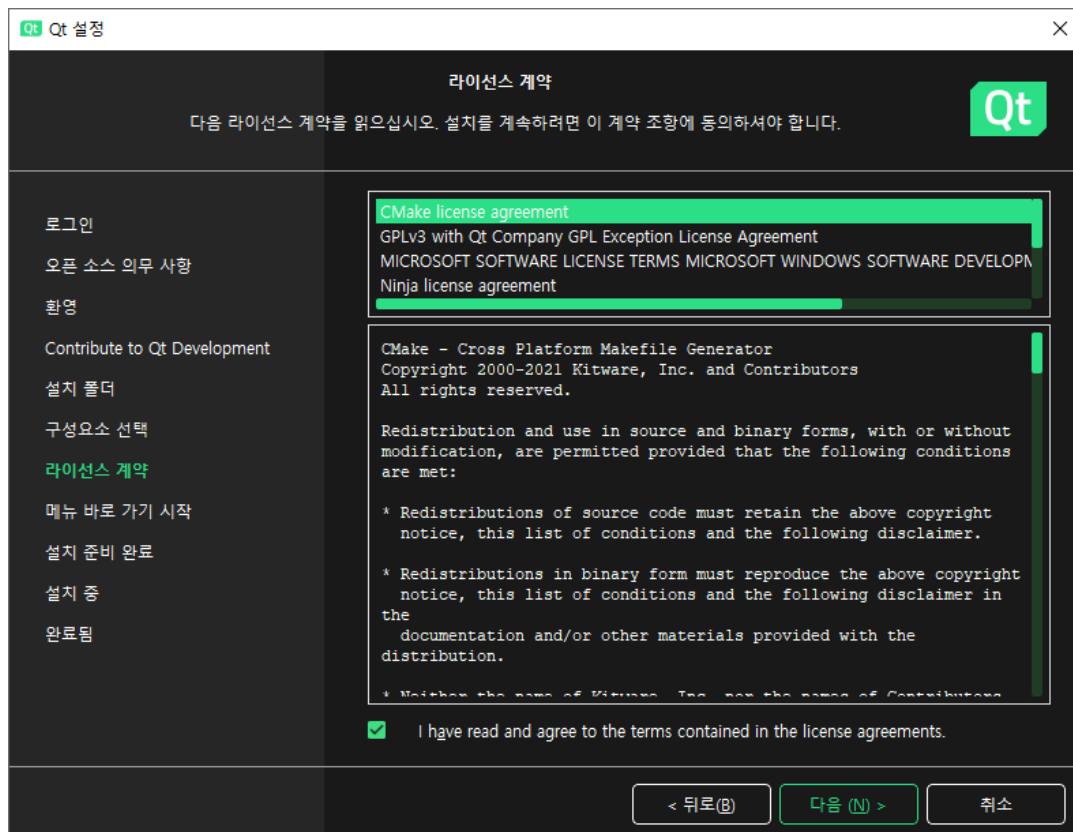
위의 화면에서 보는 것과 같이 [Custom Installation] 항목을 선택 후 [다음]을 클릭한다. 그리고 다음 화면에서는 아래와 같은 항목을 선택한다.

예수님은 당신을 사랑합니다.

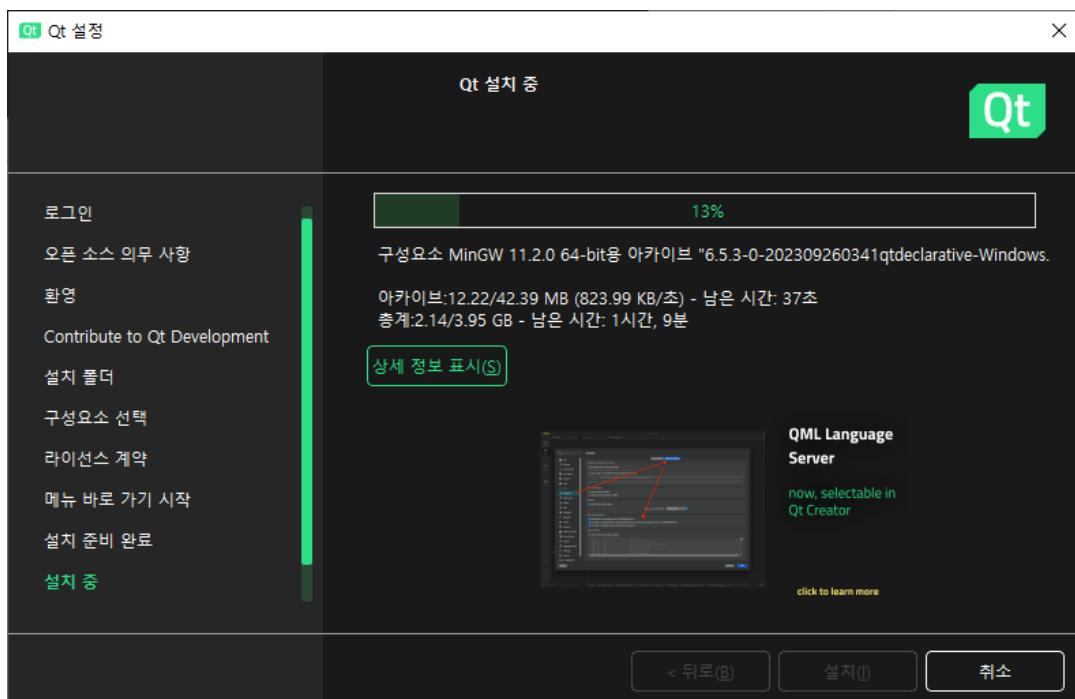
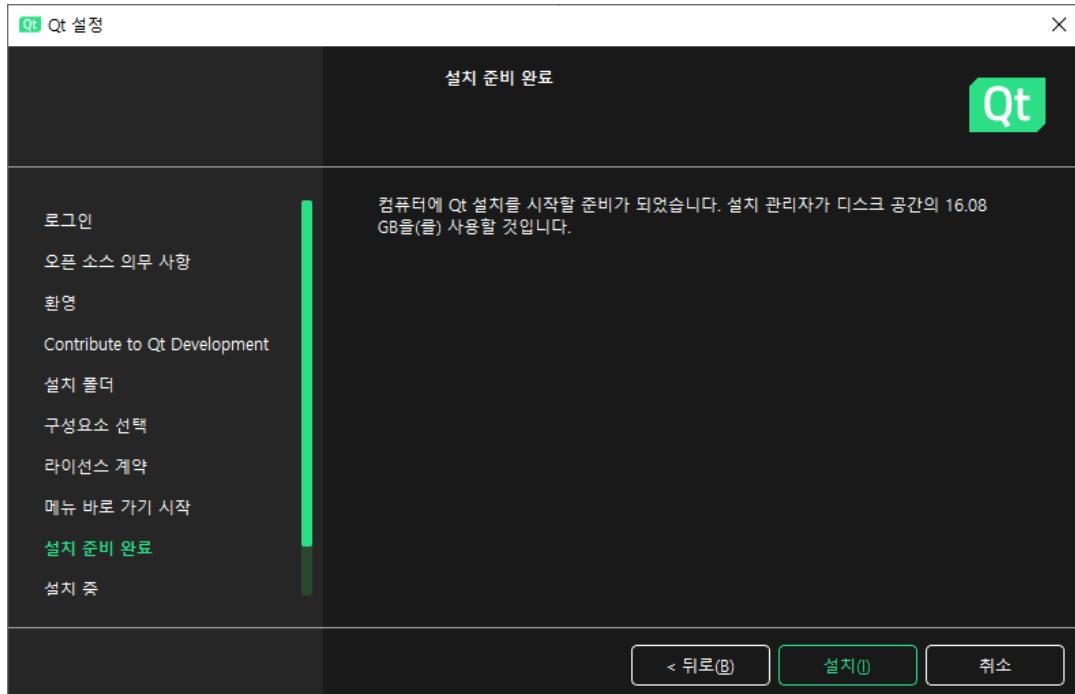


위에서 보는 것과 같이 항목을 선택한 다음 [다음] 버튼을 클릭한다.

예수님은 당신을 사랑합니다.

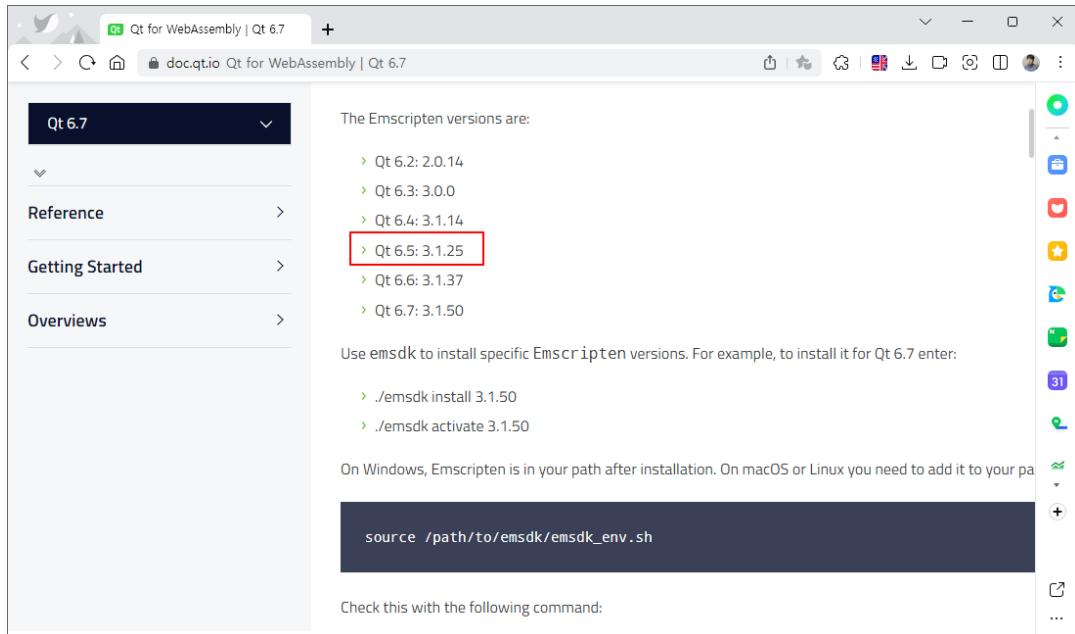


예수님은 당신을 사랑합니다.



Qt설치가 완료된 후, Qt 버전 별 Emscripten 버전을 다시 설치해야 한다. 이전에는 최신버전을 설치했지만 Qt 6.5.x LTS버전과 호환되는 Emscripten 버전을 설치해야 한다. Qt 버전별 호환되는 정보는 아래 사이트에서 확인이 가능하다.

- 버전 별 확인하는 사이트 주소: <https://doc.qt.io/qt-6/wasm.html>

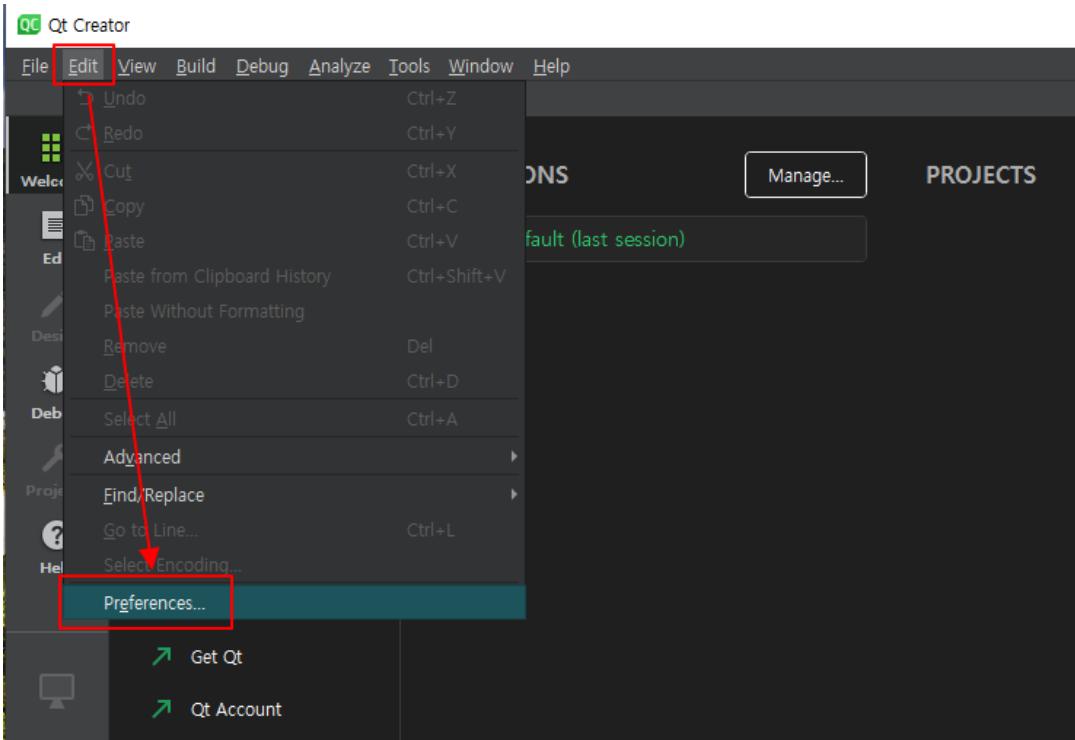


위의 그림에서 보는 것과 같이 Qt 6.5 LTS 버전과 호환되는 Emscripten 버전은 3.1.25 버전이다.

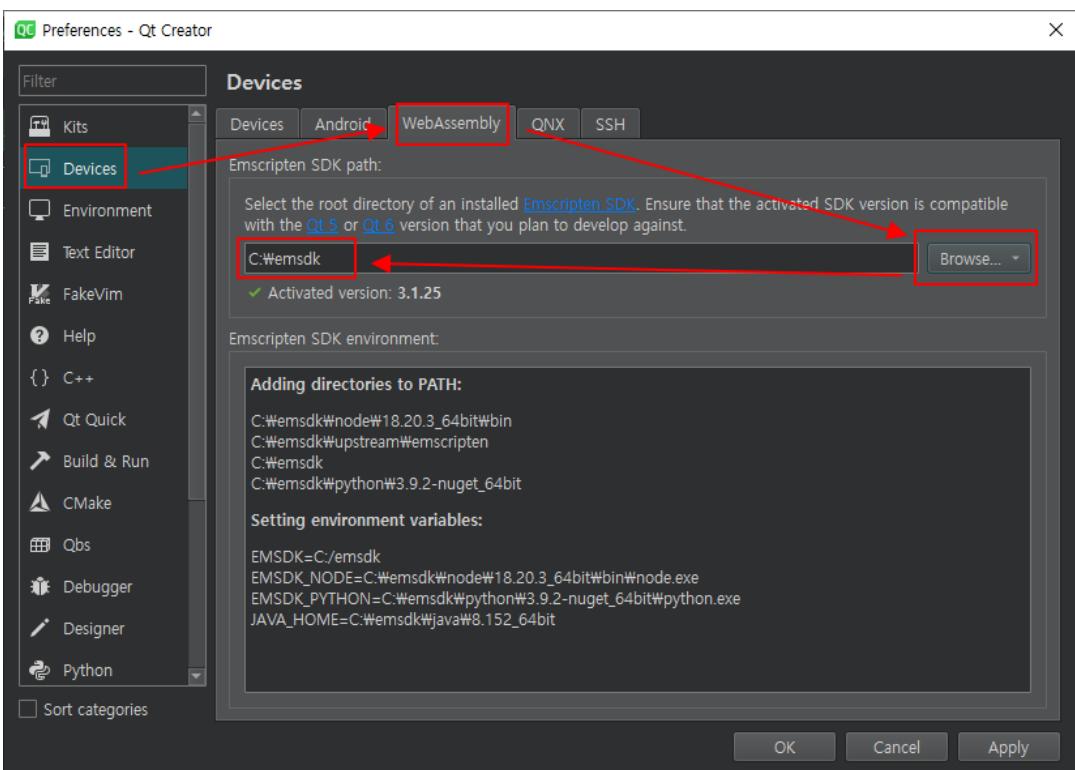
```
D:\emsdk> emsdk.bat install 3.1.25
...
D:\emsdk> emsdk.bat activate 3.1.25
...
D:\emsdk> emsdk_env.bat
...
D:\emsdk> em++ --version
emcc (Emscripten gcc/clang-like replacement + linker emulating GNU ld) 3.1.25
(fe8d44b21ecaca86e2cb2a25ef3ed4a0a2076365)
Copyright (C) 2014 the Emscripten authors (see AUTHORS.txt)
This is free and open source software under the MIT license.
There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR
PURPOSE.
```

위와 같이 버전정보에서 3.1.25버전이 포함되어 있으면 설치가 정상적으로 된 것이다.

다음으로 Qt Creator를 실행한다.



[Edit] -> [Preferences...] 메뉴를 클릭한다.



예수님은 당신을 사랑합니다.

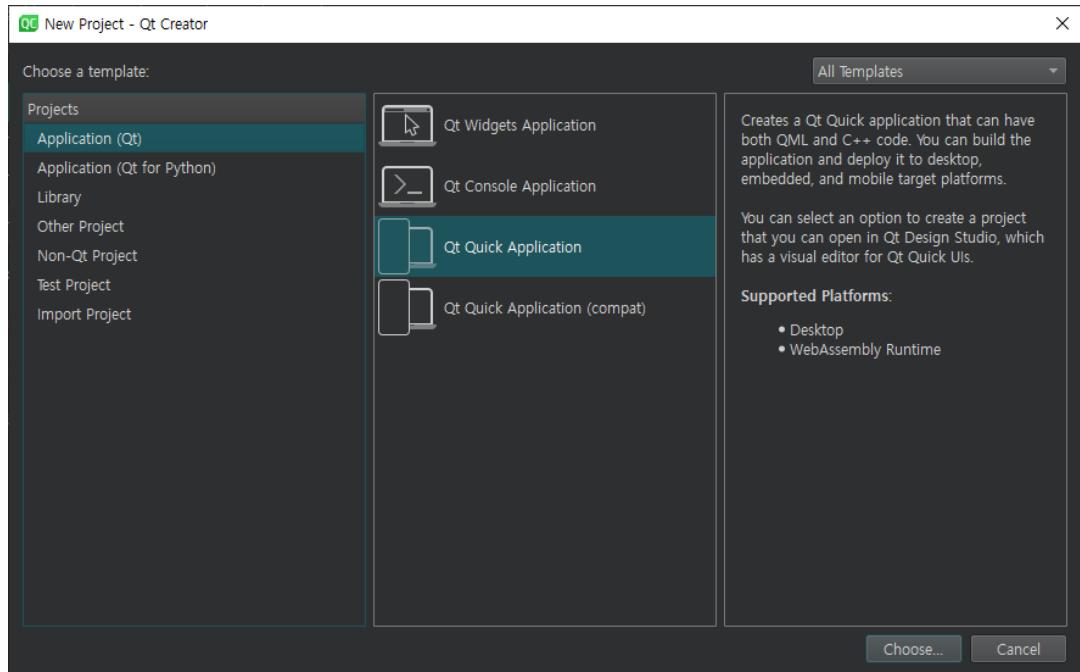
[Devices] -> [WebAssembly] -> [Browser]을 실행한다. 그리고 Emscripten이 설치된 디렉토리를 지정한다.

위와 같이 [Activated version: 3.1.25]라는 메시지를 설정하면 모든 설정이 완료된 것이다.

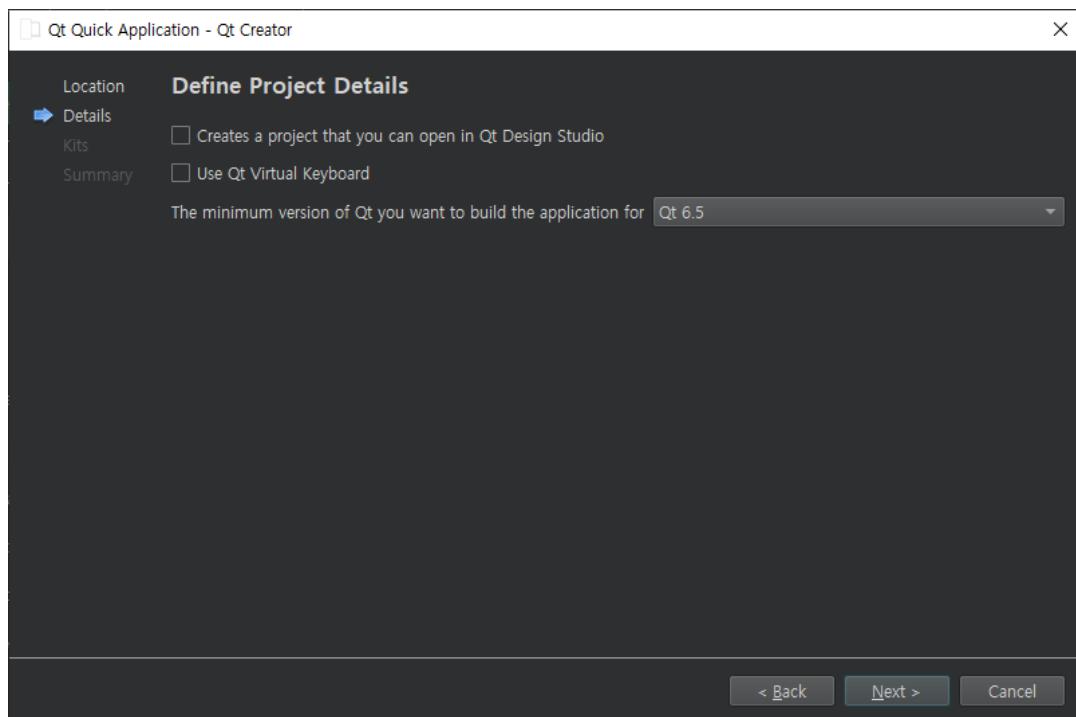
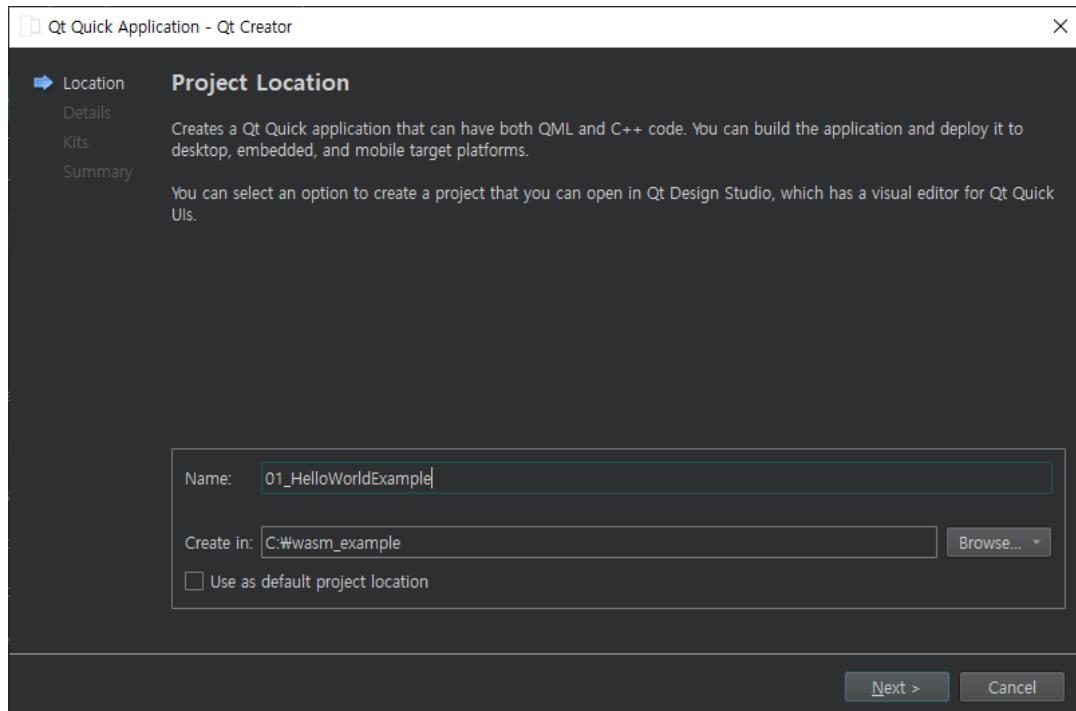
● Hello World 출력 예제

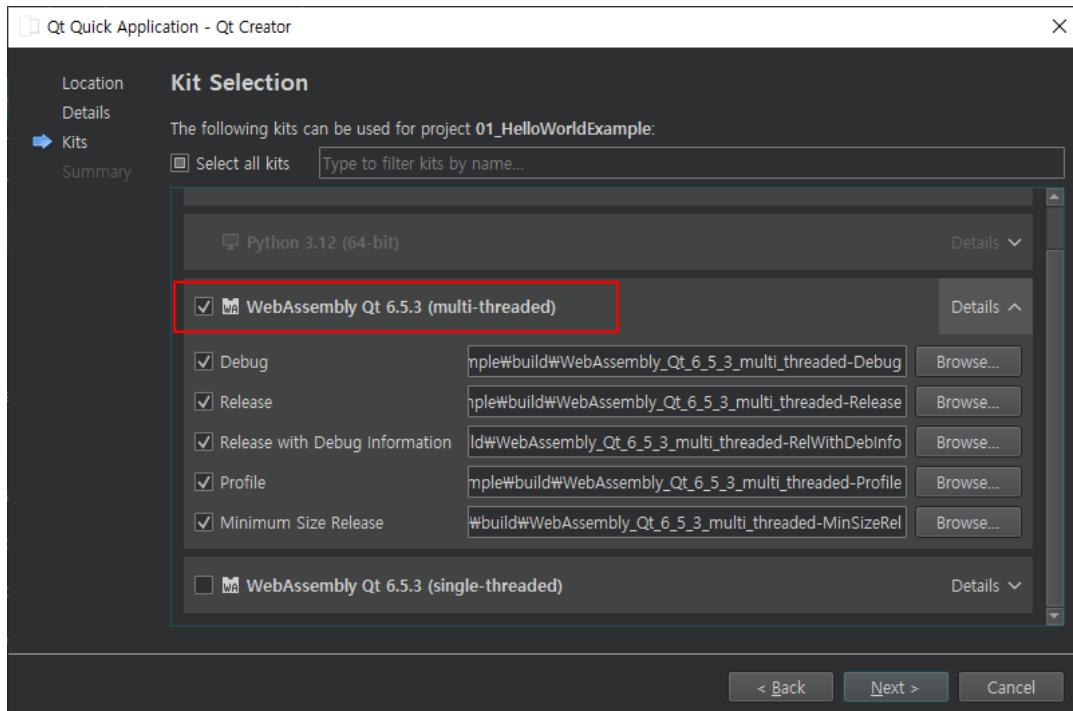
이번에는 Web Browser 상에 Hello World를 출력하는 예제를 작성해 보조 실행 보도록 하자.

Qt Creator 창에서 프로젝트를 생성한다. 아래와 같은ダイ얼로그에서 [Application Qt] 항목을 클릭 후 [Qt Quick Application]을 선택한다.

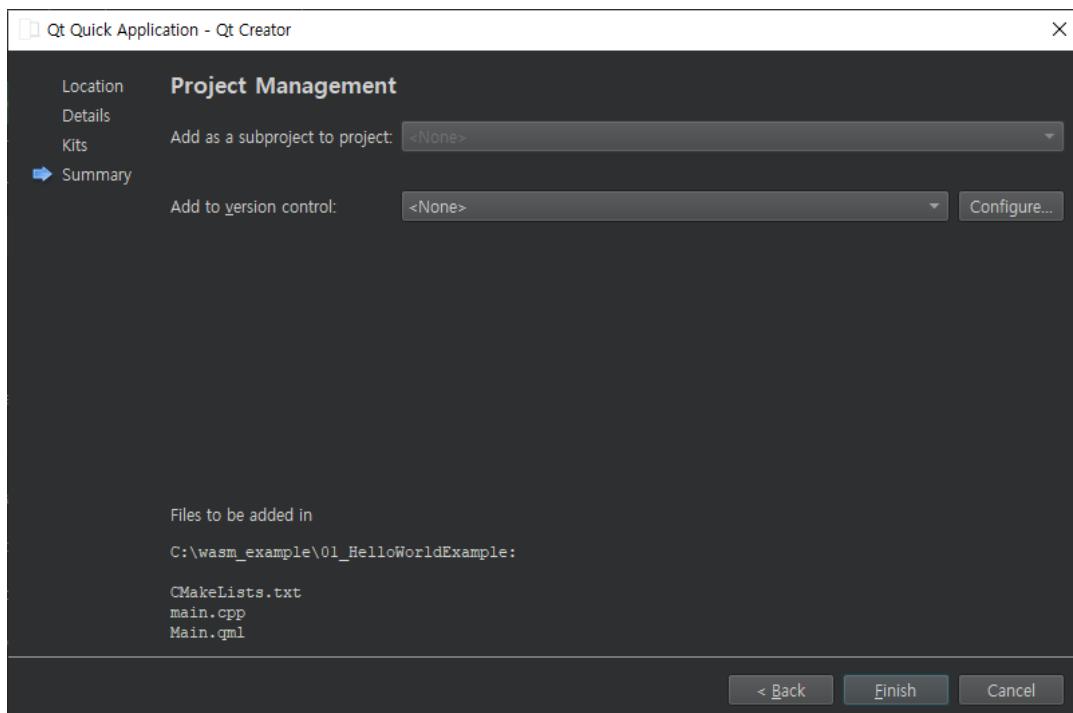


예수님은 당신을 사랑합니다.





위의 그림에서 보는 것과 같이 [WebAssembly Qt 6.5.3 (multi-threaded)] 항목을 선택한다.



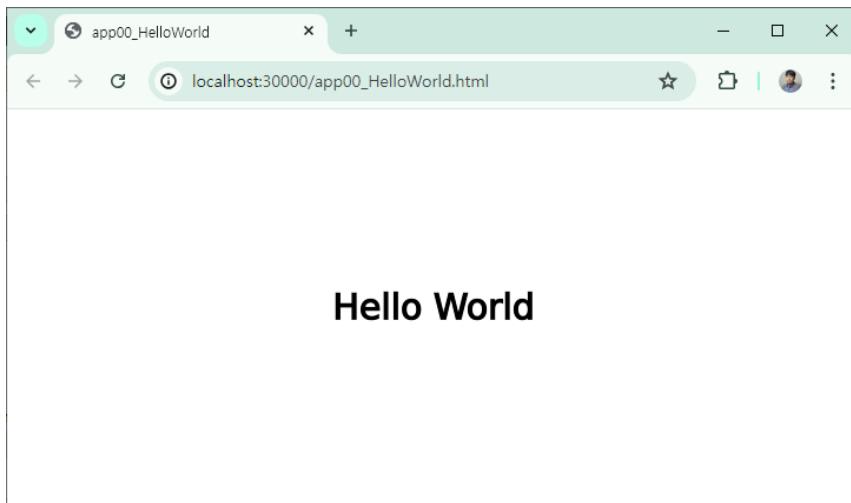
예수님은 당신을 사랑합니다.

프로젝트 생성이 완료되면 Main.qml 파일을 열어서 아래와 같이 소스코드를 작성한다.

```
import QtQuick

Window {
    width: 640; height: 480; visible: true
    Text {
        anchors.centerIn: parent
        text: "Hello World"
        font.pixelSize: 30
        font.bold: true
    }
}
```

위와 같이 소스코드를 작성한 다음 빌드 후 실행해 보도록 하자.



정상적으로 빌드가 완료되면 Web Browser가 실행되는 것을 확인할 수 있다. 빌드가 완료된 디렉토리에 가보면 WASM 파일이 생성된 것을 확인할 수 있다. 또한 WASM파일을 실행하기 위해서는 HTML파일을 만들어 주고 WASM을 실행할 Web Server 도 필요하다.

Qt Creator는 디버깅할 수 있도록 디버깅 용 Web Server를 실행해 준다. 따라서 Qt for WebAssembly는 WebAssembly 디버깅 시 필요한 요소를 편리하게 모두 제공해 준다.

3. Linux에서 개발환경 구축

리눅스에서 Qt for WebAssembly 개발 환경을 구축하기 위해서 Ubuntu Linux 20.04 LTS 버전을 사용할 것이다. 그리고 아래와 같이 Qt for Web Assembly 환경 구축에 필요한 패키지들을 먼저 설치해야 한다.

- Ubuntu Linux 설치 후 업데이트

```
$ sudo apt update
```

- g++ 과 make 설치

```
$ sudo apt install build-essential make
```

- OpenGL 설치

```
$ sudo apt install freeglut3-dev libglu1-mesa-dev mesa-common-dev
```

- libxcb 설치

```
$ sudo apt install libxcb*
```

- Python 설치

```
$ sudo apt install -y python3-pip  
$ pip3 install package_name  
$ sudo apt install -y libssl-dev libffi-dev python3-dev  
$ sudo apt install -y python3-venv  
$ pip install flask  
$ pip install pymongo dnspython
```

- Git 설치

```
$ sudo apt install git
```

- clang 설치

```
$ sudo apt install clang
```

다음으로 Emscripten SDK를 설치해야 한다. Home 디렉토리로 이동 후 아래와 같이 Git 을 이용해 다운로드 받는다.

- Emsdk 다운로드 및 설치

```
$ git clone https://github.com/emscripten-core/emsdk.git
```

위와 같이 clone이 완료되면 다음과 같이 Emscripten SDK를 설치한다.

```
$ cd emsdk  
$ git pull  
$ ./emsdk install 3.1.25  
$ ./emsdk activate 3.1.25  
$ source ./emsdk_env.sh
```

다음으로 emcc를 이용해 설치가 정상적으로 되었는지 버전 정보를 출력해보자.

```
$ emcc -v  
emcc (Emscripten gcc/clang-like replacement + linker emulating GNU ld) 3.1.25  
(febcd44b21ecaca86e2cb2a25ef3ed4a0a2076365)  
clang version 16.0.0 (https://github.com/llvm/llvm-project)  
Target: wasm32-unknown-emscripten  
Thread model: posix  
InstalledDir: /home/dev/emsdk/upstream/bin
```

- 부팅 시 자동실행 되도록 스크립트 작업

리눅스에서 부팅 시, emsdk 스크립트가 자동으로 실행되게 하기 위해서 아래와 같이 수정한다.

예수님은 당신을 사랑합니다.

```
$ sudo vi /etc/profile
```

다음으로 HOME/.profile 파일에 아래와 같은 라인을 추가한다.

```
source /home/dev/emsdk/emsdk_env.sh
```

다음으로 Hello world를 출력하는 예제를 작성한후, 직업 Emscripten SDK를 사용해 빌드 후 Web Browser에서 실행해 보도록 하자.

hello.c 소스코드 파일을 만든 후 아래와 같이 소스코드를 추가한다.

```
#include <stdio.h>
int main()
{
    printf("Hello World ~ ^~; \n");
    return 1;
}
```

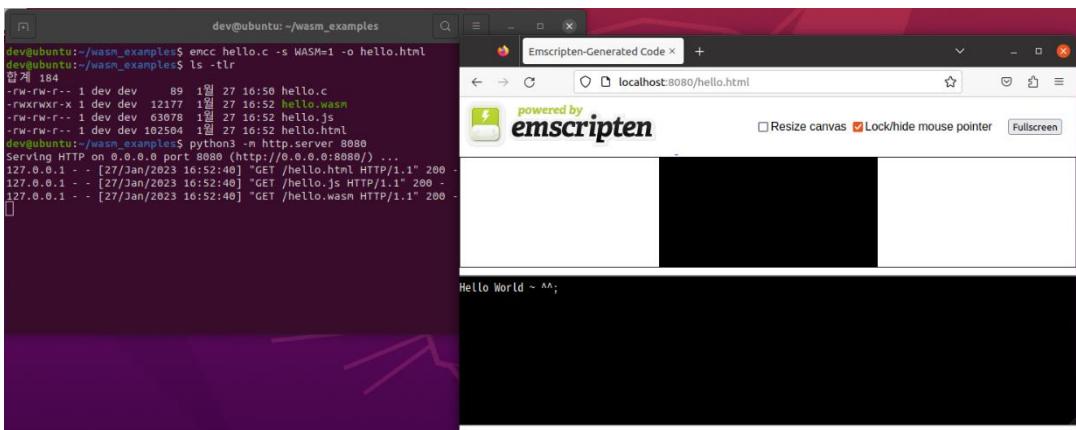
다음으로 아래와 같은 방법으로 빌드 후, Python에서 제공하는 웹 서버를 로딩한다.

```
$ emcc hello.c -s WASM=1 -o hello.html
shared:INFO: (Emscripten: Running sanity checks)
cache:INFO: generating system asset: ...
cache:INFO: - ok
$ ls -tlr
합계 448
-rw-rw-r-- 1 dev dev    477 7월 21 00:17 hello.cpp
-rwxrwxr-x 1 dev dev 158679 7월 21 00:19 hello.wasm
-rw-rw-r-- 1 dev dev 184873 7월 21 00:19 hello.js
-rw-rw-r-- 1 dev dev 102507 7월 21 00:19 hello.html
```

```
$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
```

다음으로 Web Browser를 이용해 <http://localhost:8080/hello.html> URL을 실행해 보도록 하자.

예수님은 당신을 사랑합니다.



```
dev@ubuntu:~/wasm_examples$ emcc hello.c -s WASM=1 -o hello.html
dev@ubuntu:~/wasm_examples$ ls -lR
합계 184
-rw-rw-r-- 1 dev dev 89 1월 27 16:50 hello.c
-rwxrwxr-x 1 dev dev 12177 1월 27 16:52 hello.wasm
-rw-rw-r-- 1 dev dev 63078 1월 27 16:52 hello.js
-rw-rw-r-- 1 dev dev 10258 1월 27 16:52 hello.html
dev@ubuntu:~/wasm_examples$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
127.0.0.1 - - [27/Jan/2023 16:52:40] "GET /hello.html HTTP/1.1" 200 -
127.0.0.1 - - [27/Jan/2023 16:52:40] "GET /hello.js HTTP/1.1" 200 -
127.0.0.1 - - [27/Jan/2023 16:52:40] "GET /hello.wasm HTTP/1.1" 200 -
```

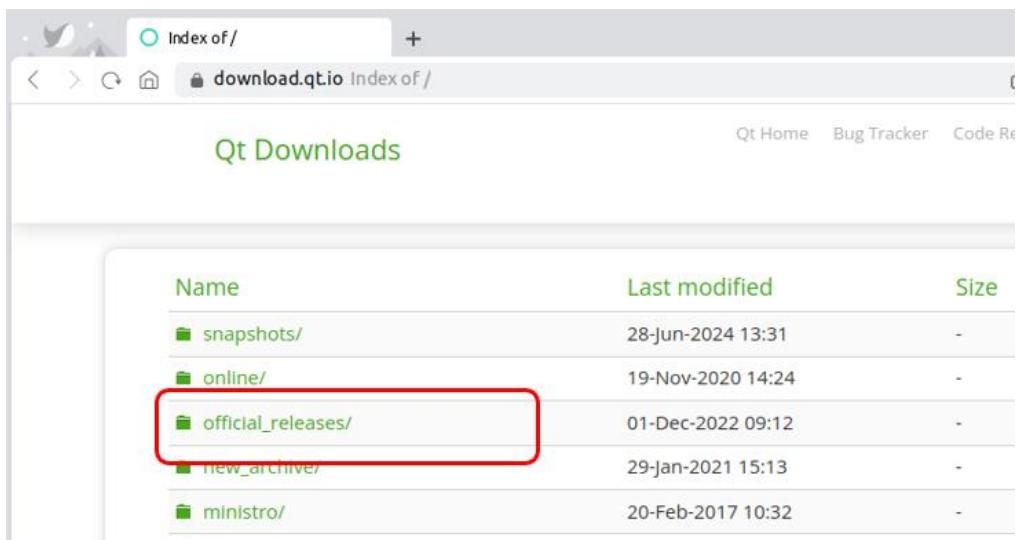
powered by **emscripten** Resize canvas Lock/hide mouse pointer Fullscreen

Hello World ~ ^~;

다음으로 Qt Framework 를 설치해 보도록 하자.

● Qt Framework 설치

Qt Framework를 설치하기 위해서 온라인 설치파일 다운로드 받는다. 온라인 설치 파일은 <https://download.qt.io> 사이트 URL에서 다운로드 받을 수 있다.



Name	Last modified	Size
snapshots/	28-Jun-2024 13:31	-
online/	19-Nov-2020 14:24	-
official_releases/	01-Dec-2022 09:12	-
new_archive/	29-Jan-2021 15:13	-
ministro/	20-Feb-2017 10:32	-

위의 그림에서 보는 것과 같이 [official_releases] 링크를 클릭한다. 다음으로 아래 그림에서 보는 것과 같이 [online_installers] 링크를 클릭한다.

예수님은 당신을 사랑합니다.

The screenshot shows a list of files under the 'online_installers' directory:

Name	Last modified	Size	Metadata
Parent Directory	-	-	
vsaddin/	22-Apr-2024 10:31	-	
qtdesignstudio/	13-Jul-2022 20:21	-	
qtcreator/	07-Jun-2024 12:25	-	
qtchooser/	08-Oct-2018 07:53	-	
qt3dstudio/	28-Oct-2020 14:22	-	
qt/	09-Apr-2024 10:25	-	
qt-installer-framework/	15-May-2024 11:04	-	
qbs/	07-May-2024 16:44	-	
pyside/	30-Nov-2015 13:39	-	
online_installers/	15-May-2024 12:08	-	
jom/	05-Sep-2023 15:36	-	
gdb/	17-Nov-2014 13:42	-	
additional_libraries/	03-Mar-2021 10:18	-	
QtForPython/	08-Mar-2024 15:29	-	
timestamp.txt	01-Jul-2024 08:00	11	Details

다음으로 아래 그림에서 보는 것과 같이 확장자가 qt-online-installer-linux-x64-4.8.0.run 파일을 다운로드 받는다.

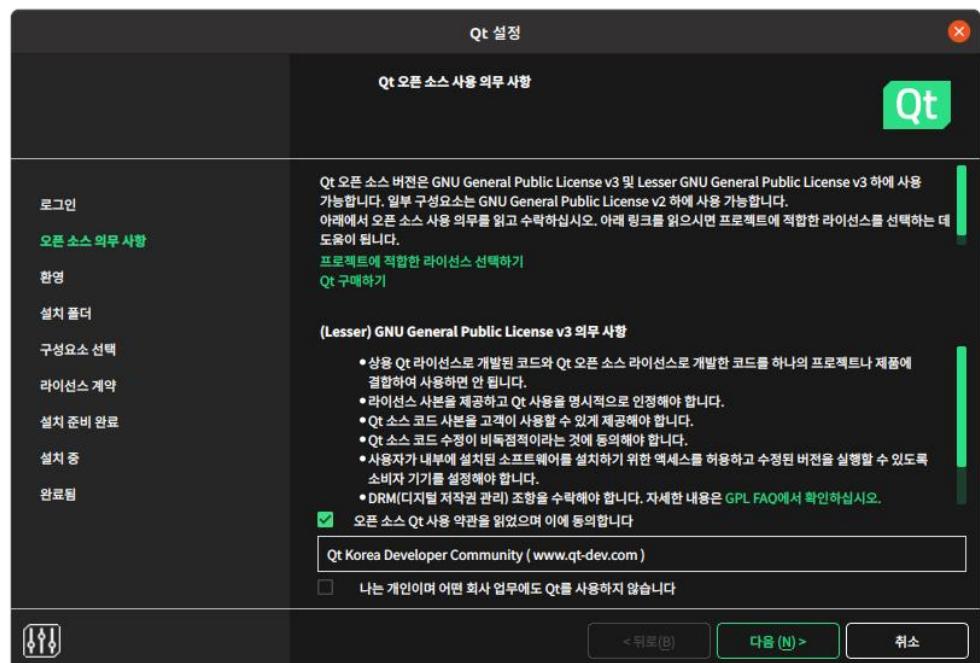
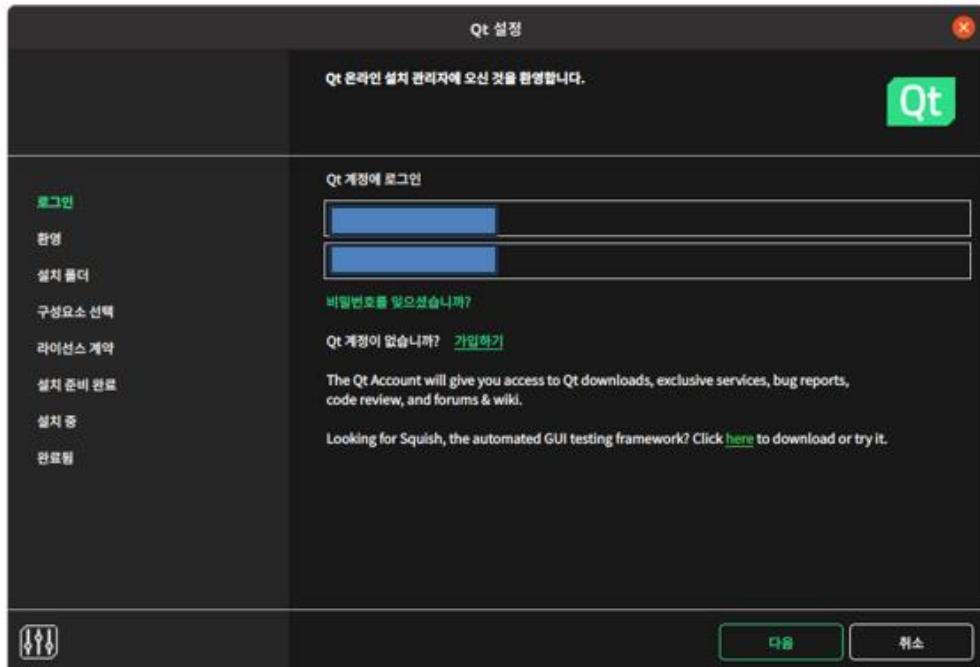
The screenshot shows a list of files under the 'online_installers' directory:

Name	Last modified
Parent Directory	
qt-unified-windows-x64-online.exe	15-May-2024 10:07
qt-unified-mac-x64-online.dmg	15-May-2024 10:07
qt-unified-linux-x64-online.run	15-May-2024 10:07
qt-unified-linux-arm64-online.run	15-May-2024 10:07

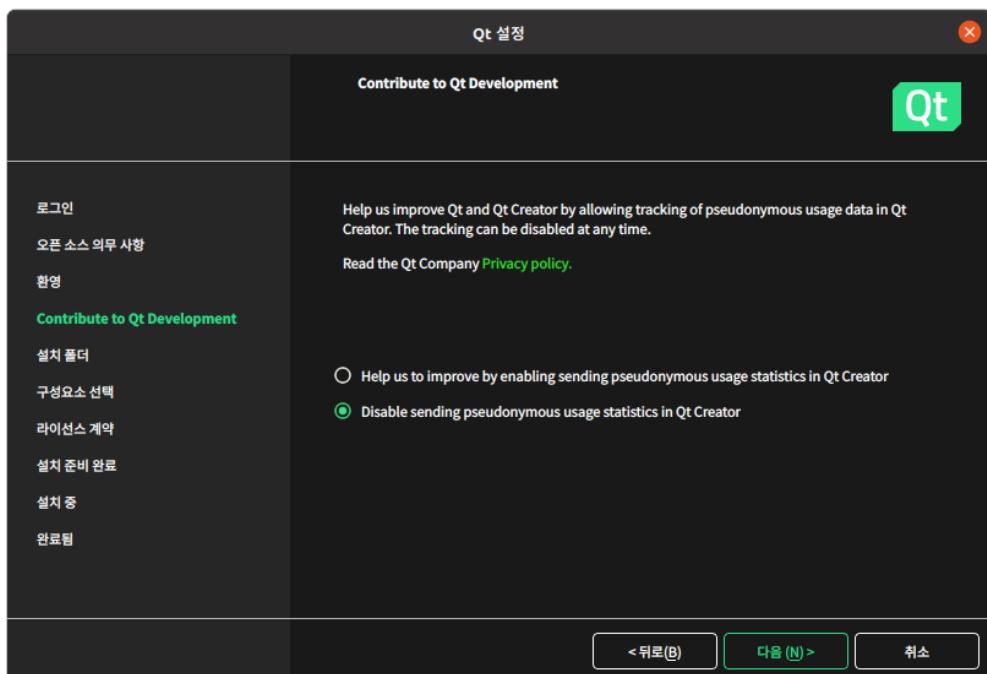
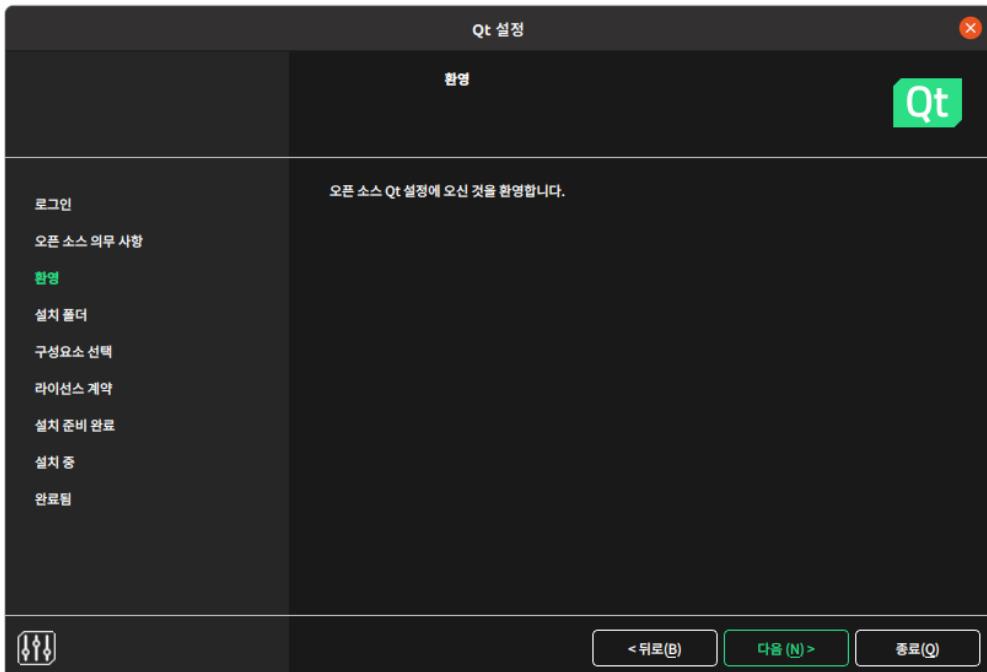
qt-online-installer-linux-x64-4.8.0.run 파일 다운로드가 완료되면, 이 파일의 실행권한을 부여한다. 그리고 실행한다.

```
$ chmod 755 qt-online-installer-linux-x64-4.8.0.run  
$ ./qt-online-installer-linux-x64-4.8.0.run
```

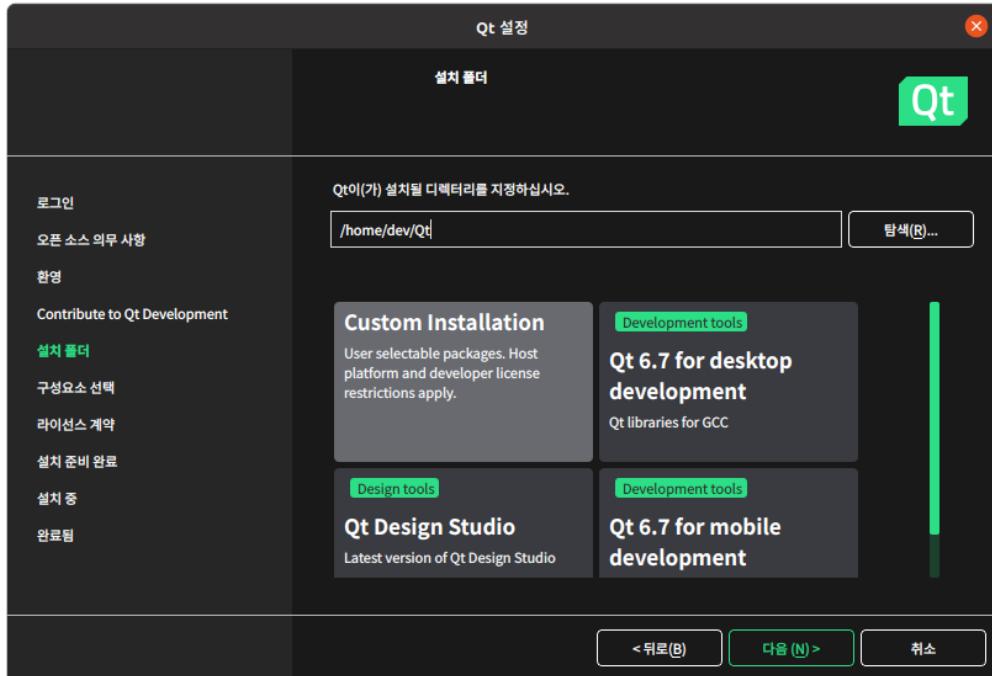
예수님은 당신을 사랑합니다.



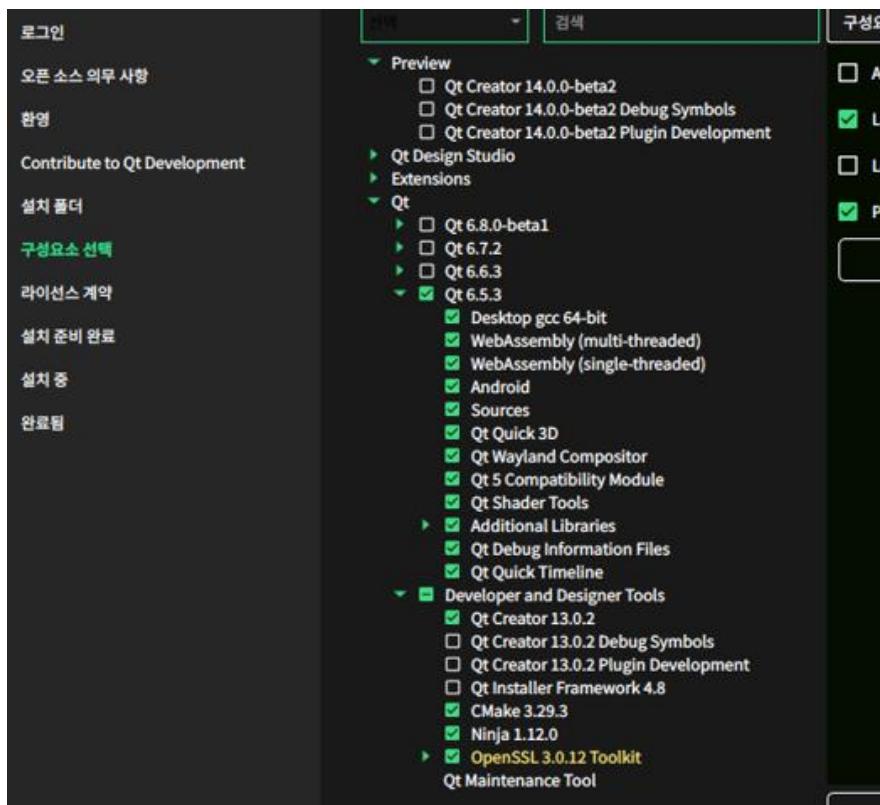
예수님은 당신을 사랑합니다.



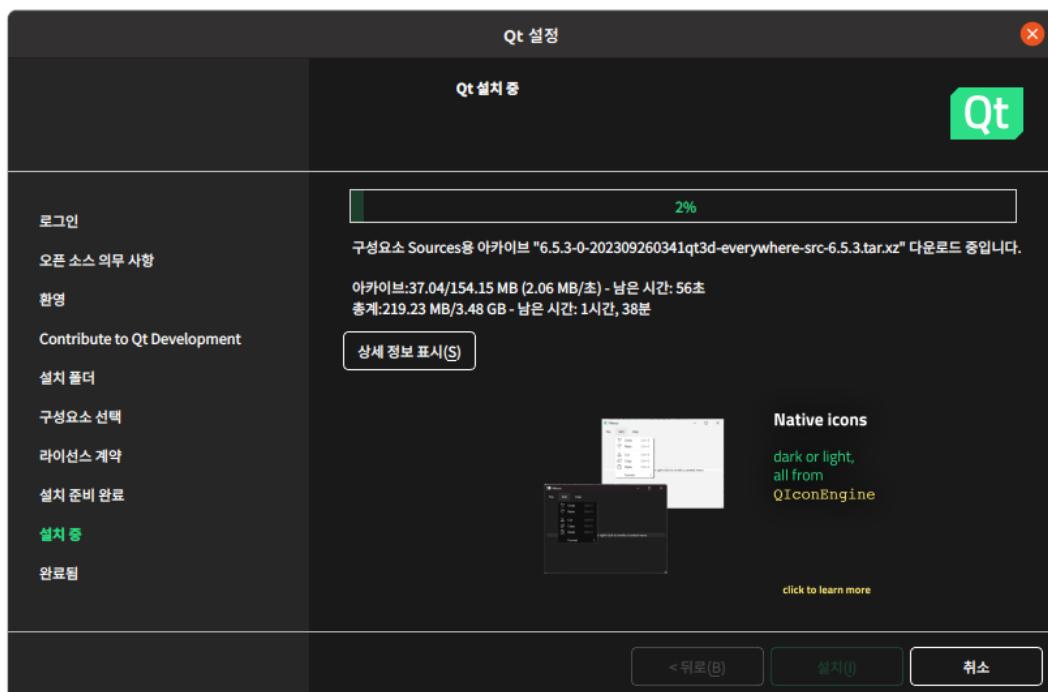
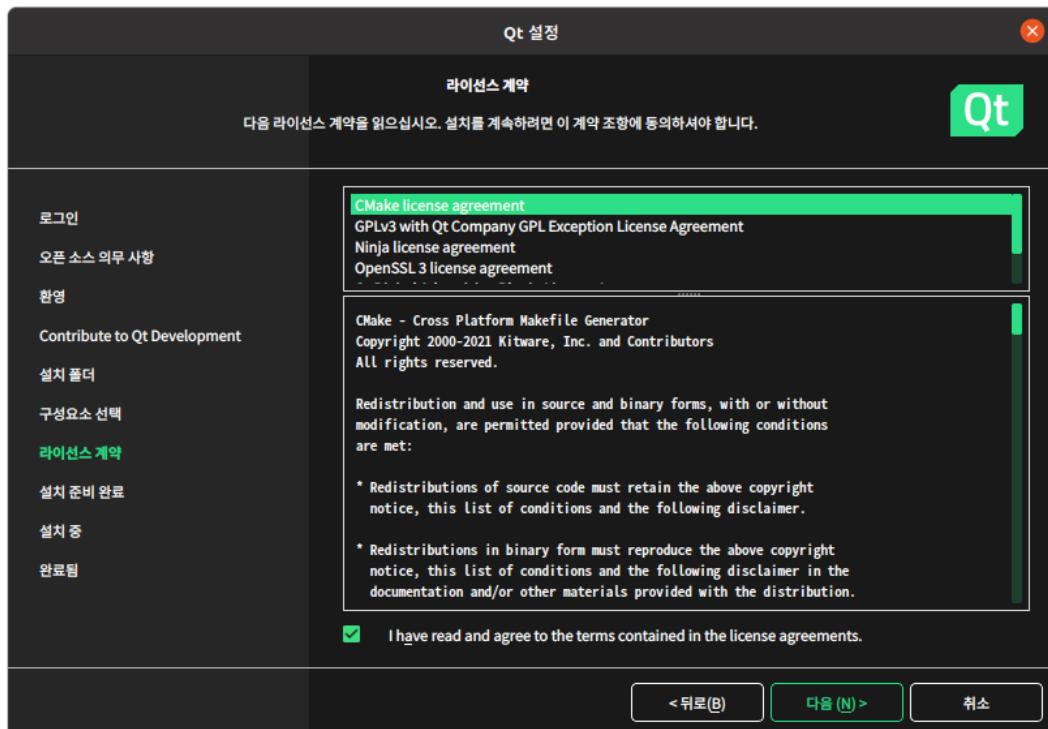
예수님은 당신을 사랑합니다.



위의 그림에서 보는 것과 같이 [Custom Installation] 항목을 선택한다. 다음 다이얼로그에서는 Qt 6.5 버전을 선택하고 아래 그림과 같이 항목들을 선택한다.



예수님은 당신을 사랑합니다.



예수님은 당신을 사랑합니다.

Qt설치가 완료되면 Emscripten버전이 Qt 6.5버전과 호환되는 버전인지 아래와 같이 확인해본다.

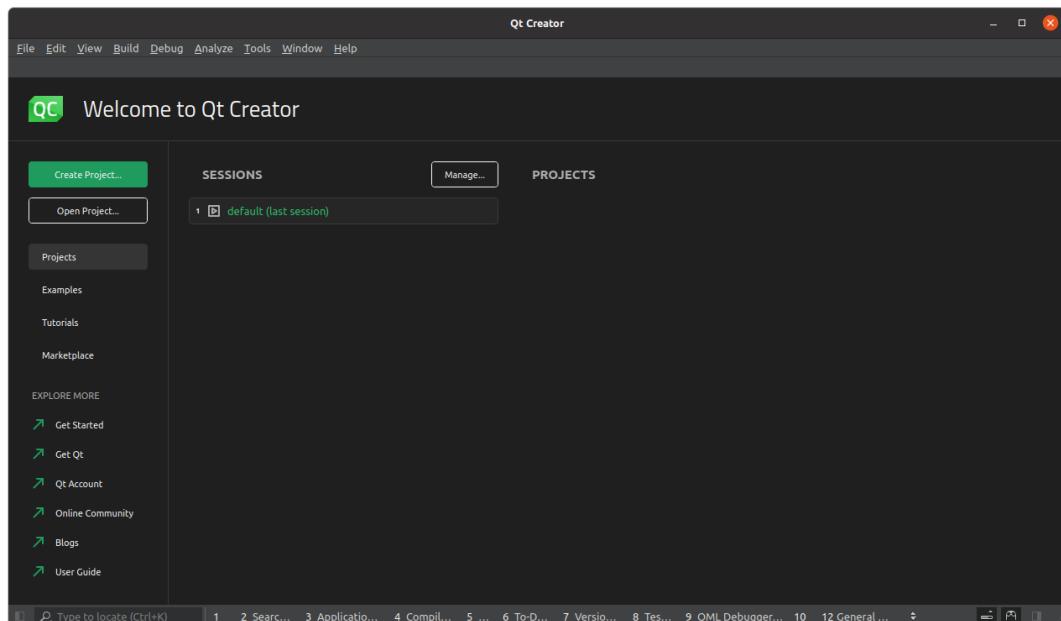
```
$ source /home/dev/emscripten/emsdk/emsdk_env.sh
$ em++ --version
emcc (Emscripten gcc/clang-like replacement + linker emulating GNU ld) 3.1.25
(4343cbec72b7db283ea3bda1adc6cb1811ae9a73)
Copyright (C) 2014 the Emscripten authors (see AUTHORS.txt)
This is free and open source software under the MIT license.
There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR
PURPOSE.
```

Qt버전은 되도록 LTS버전을 사용하는 것을 추천한다. 만약 다른 Qt버전을 사용하는 경우 아래 URL을 참고하면 된다.

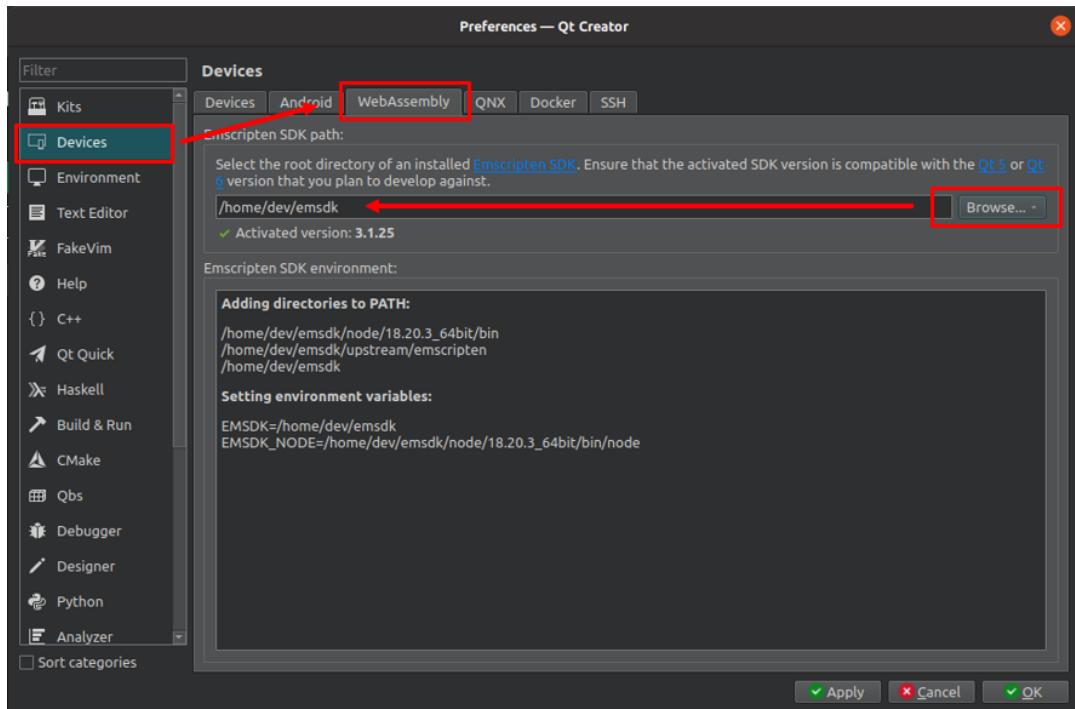
- <https://doc.qt.io/qt-6/wasm.html>

● Qt WebAssembly 설정

Qt Creator를 실행한다.

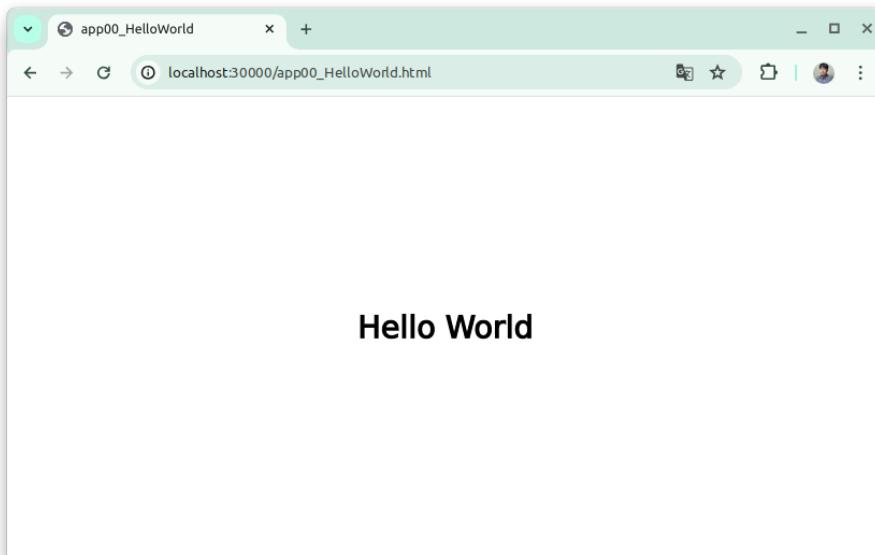


Qt Creator 실행 후 [Edit] 메뉴를 클릭한 후 [Preferences...] 메뉴를 클릭한다.



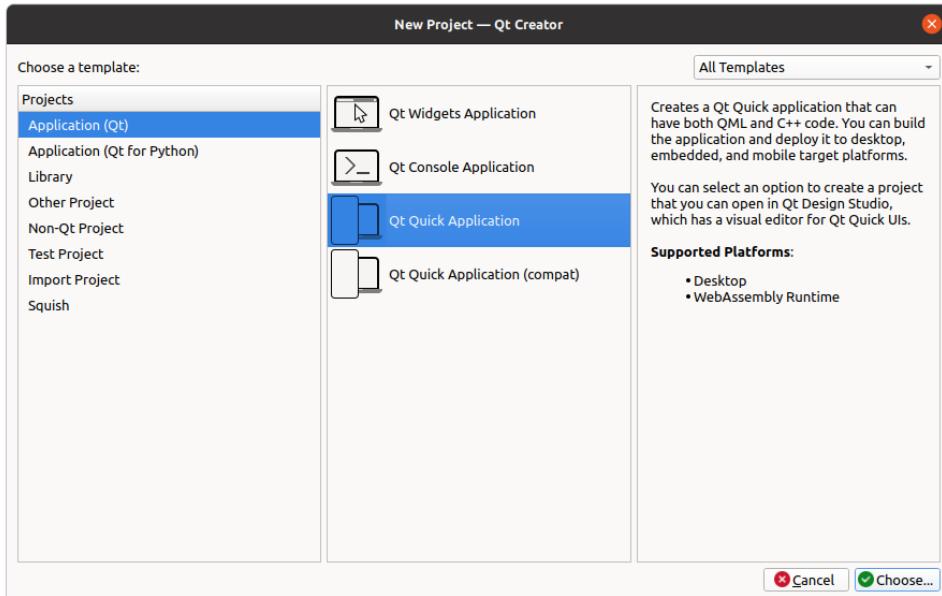
[Preferences] 다이얼로그의 왼쪽 Tab에서 [Devices]를 선택한다. 그리고 우측의 Tab에서 [WebAssembly] 항목을 선택한다. 다음으로 [Browse] 버튼을 클릭하고 Emscripten SDK 가 설치된 디렉토리를 선택한다.

- Hello World를 Web Browser에 출력하는 예제

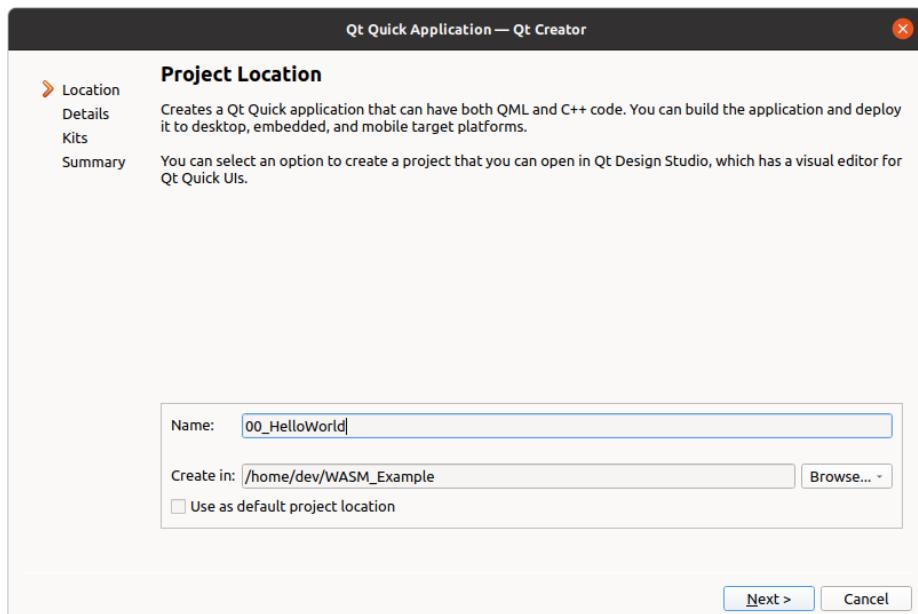


예수님은 당신을 사랑합니다.

위의 그림에서 보는 것과 같이 Web Browser에 "Hello World"를 출력하는 예제를 작성해 보도록 한다. 아래와 같이 프로젝트를 생성해 보도록 하자.

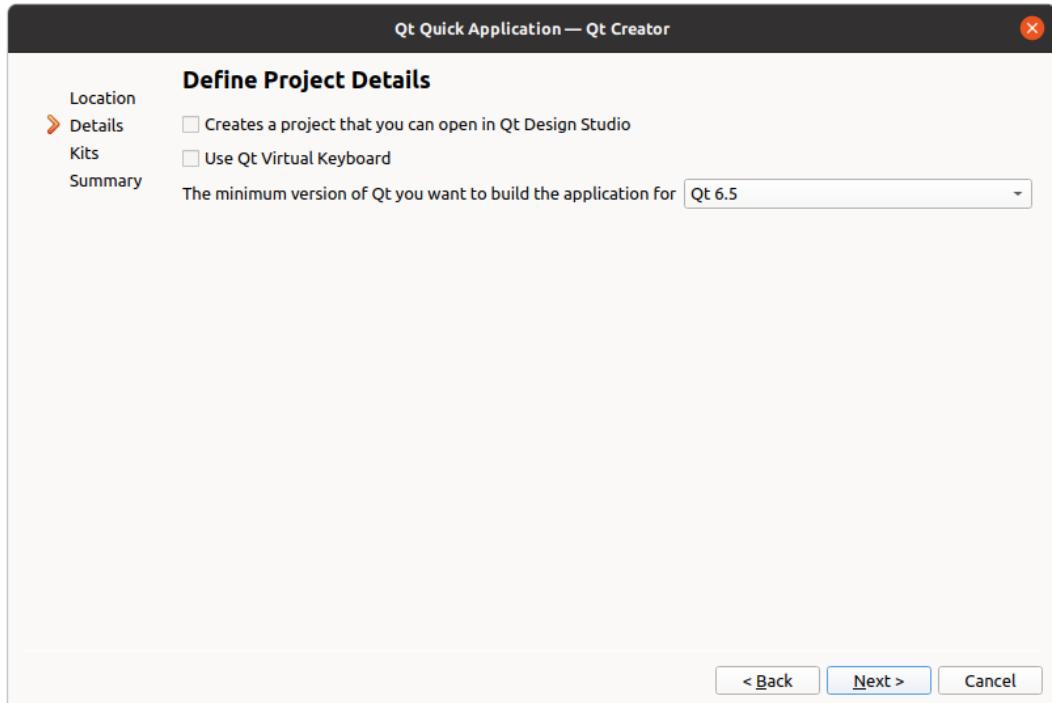


위의 그림에서 보는 것과 같이 [Application (Qt)] 항목을 선택한다. 그리고 중간에 위치한 항목 중에서 [Qt Quick Application] 항목을 선택한다. 그리고 하다는 [Choose] 버튼을 클릭한다.

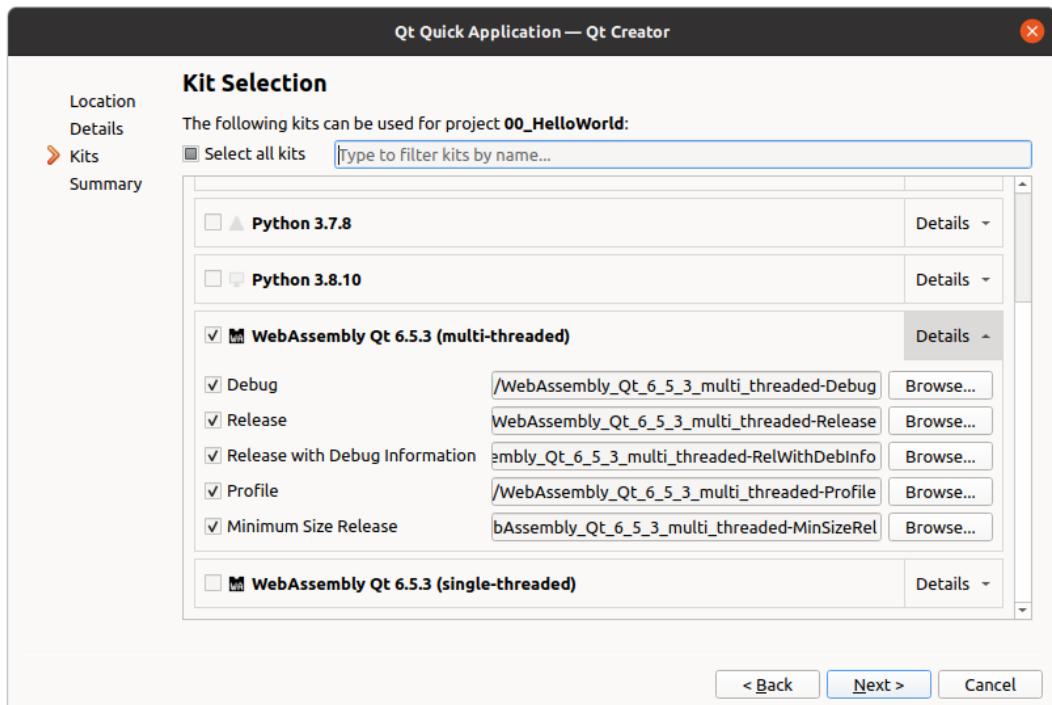


위의 그림에서 보는 것과 같이 프로젝트 이름을 입력하고 [Next] 버튼을 클릭한다.

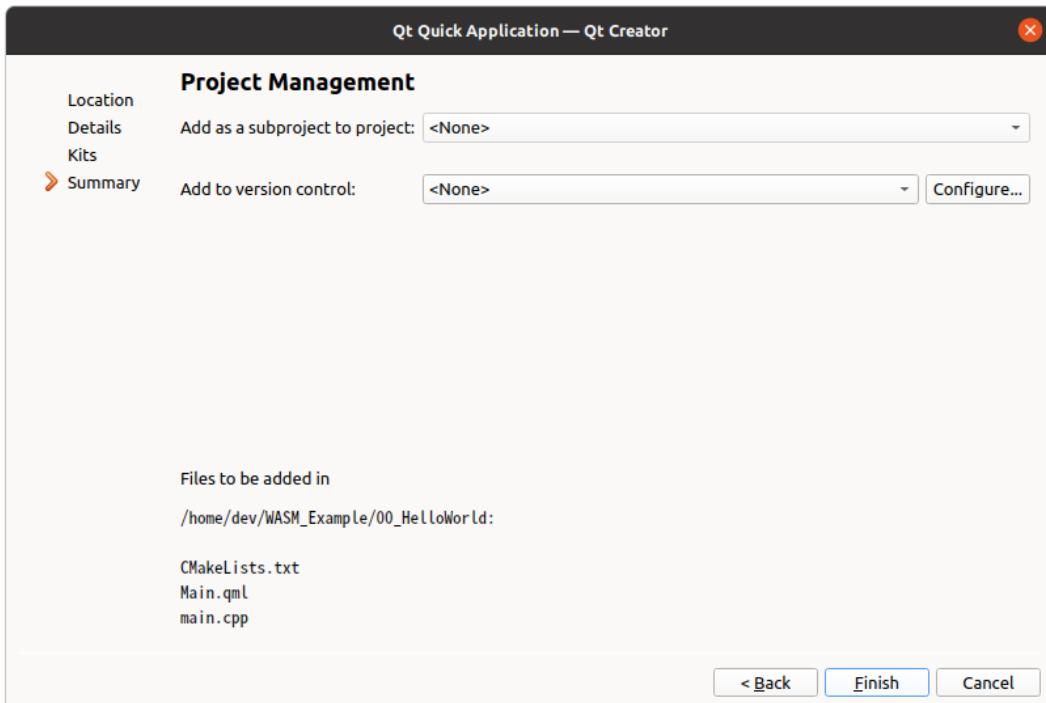
예수님은 당신을 사랑합니다.



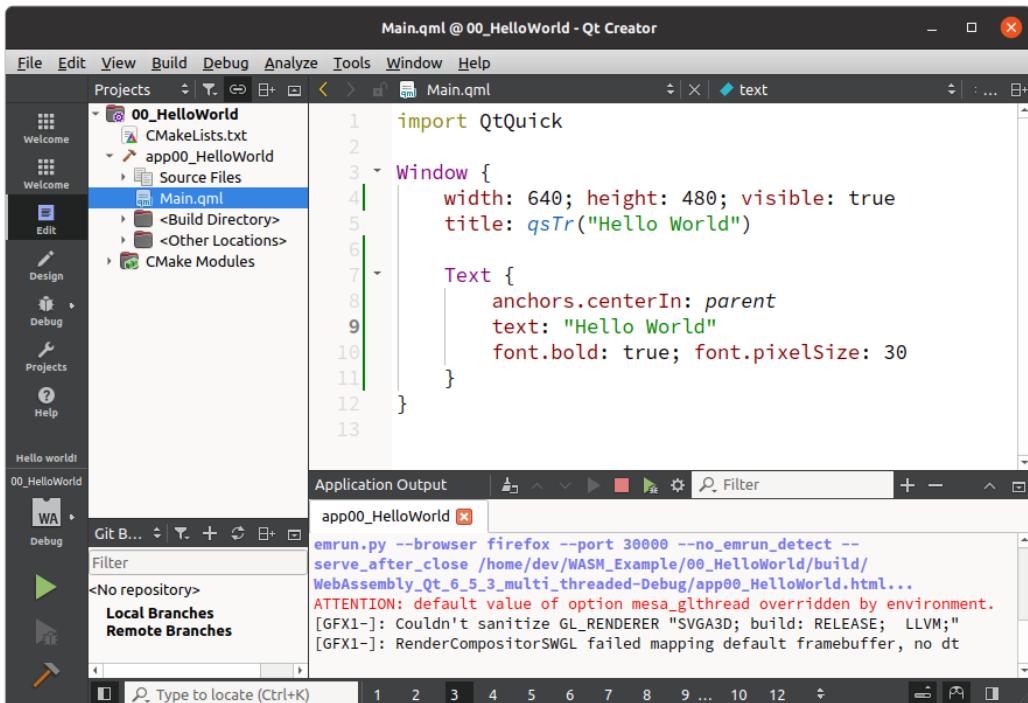
Qt의 Minimum Version으로 Qt 6.5버전을 선택한다. 그리고 아래 그림에서 보는 것과 같이 [Kits] 항목에서 [WebAssembly Qt 6.x.x (multi-threaded)] 항목을 선택한다.



예수님은 당신을 사랑합니다.



[Finish]버튼을 클릭하면 프로젝트 생성이 완료된다. 다음으로 Main.qml 파일을 열어서 다음과 같이 소스코드를 작성한다.



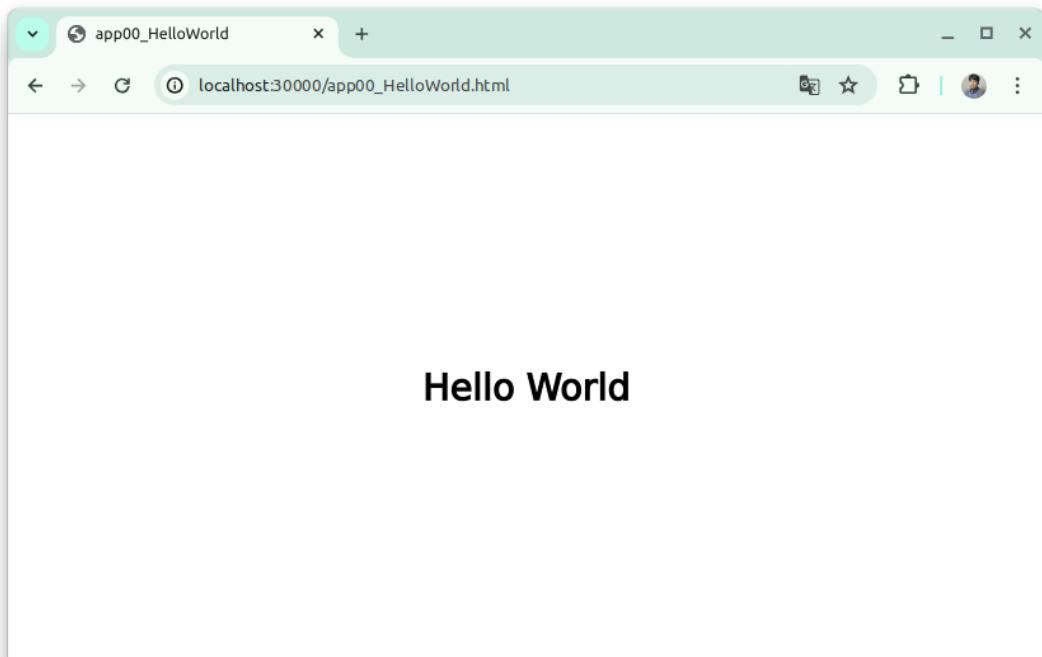
예수님은 당신을 사랑합니다.

```
import QtQuick

Window {
    width: 640; height: 480; visible: true
    title: qsTr("Hello World")

    Text {
        anchors.centerIn: parent
        text: "Hello World"
        font.bold: true; font.pixelSize: 30
    }
}
```

위와 같이 소스코드를 작성한 다음 빌드 후 실행해 보도록 하자. 그러면 아래와 같이 Web Browser 상에 "Hello World" 가 출력되는 것을 확인할 수 있을 것이다.



4. macOS에서 개발환경 구축

macOS에서 Qt WebAssembly 개발 환경을 구축하기 위해서, 먼저 아래와 같은 패키지를 먼저 설치해야 한다.

- Homebrew설치

```
% /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

- Homebrew설치 완료 후 버전 확인

```
% brew --version
```

- Homebrew로 Python3설치

```
% brew install python3
```

- Python3설치 후 버전 확인

```
% python3 --version
```

```
Python 3.x.x
```

- cmake 설치

```
% brew install cmake
```

- Emscripten 다운로드 및 설치

```
% git clone https://github.com/juj/emsdk.git
```

```
% cd emsdk
```

```
emsdk % ./emsdk update
```

예수님은 당신을 사랑합니다.

```
emsdk % git pull  
emsdk % ./emsdk install 3.1.14  
emsdk % ./emsdk activate 3.1.14  
emsdk % source ./emsdk_env.sh
```

- Emscripten 버전 확인

```
emsdk % emcc --version  
emcc (Emscripten gcc/clang-like replacement + linker emulating GNU ld) 3.1.14  
(cfe2bdfe2692457cb5f5770672f6e5ccb3ffc2f2)  
Copyright (C) 2014 the Emscripten authors (see AUTHORS.txt)  
This is free and open source software under the MIT license.  
There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR  
PURPOSE.
```

- Hello world를 출력하는 예제

이번에는 Emscripten SDK를 이용해 간단히 Hello World를 출력하는 예제를 작성해 보도록 하자. 아래와 같이 디렉토리를 생성하고, 생성한 디렉토리로 이동하자.

```
% mkdir wasm_examples  
% cd wasm_examples  
wasm_examples % mkdir hello  
wasm_examples % cd hello  
dev@devui-Mac hello % pwd  
/Users/dev/wasm_examples/hello
```

다음으로 hello.c 파일을 생성한다.

```
hello % touch hello.c  
hello % ls -tlr  
total 0  
-rw-r--r-- 1 dev staff 0 1 22 19:42 hello.c  
dev@devui-Mac hello %
```

예수님은 당신을 사랑합니다.

생성한 hello.c 파일에 아래와 같이 소스코드를 작성한다.

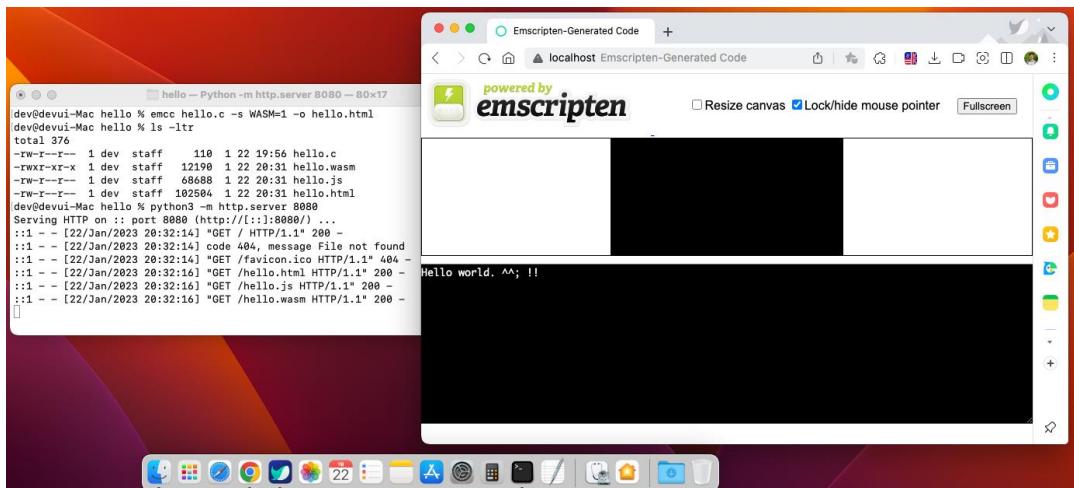
```
#include <stdio.h>
int main(int argc, char **argv)
{
    printf("Hello world. ^^; !!\n");
    return 1;
}
```

작성한 hello.c 소스코드 파일을 Emscripten SDK를 이용해 아래와 같이 빌드해 보자.

```
hello % emcc hello.c -s WASM=1 -o hello.html
hello % ls -tlr
total 376
-rw-r--r-- 1 dev staff 110 1 22 19:56 hello.c
-rwxr-xr-x 1 dev staff 12190 1 22 20:00 hello.wasm
-rw-r--r-- 1 dev staff 68688 1 22 20:00 hello.js
-rw-r--r-- 1 dev staff 102504 1 22 20:00 hello.html

dev@devui-Mac hello % python3 -m http.server 8080
Serving HTTP on :: port 8080 (http://[::]:8080) ...
```

Python에는 WASM을 실행할 수 있는 웹서버를 제공한다. 위와 같이 8080 포트 번호로 웹서버를 로딩 하면 Web Browser로 WebAssembly 파일을 실행할 수 있다.



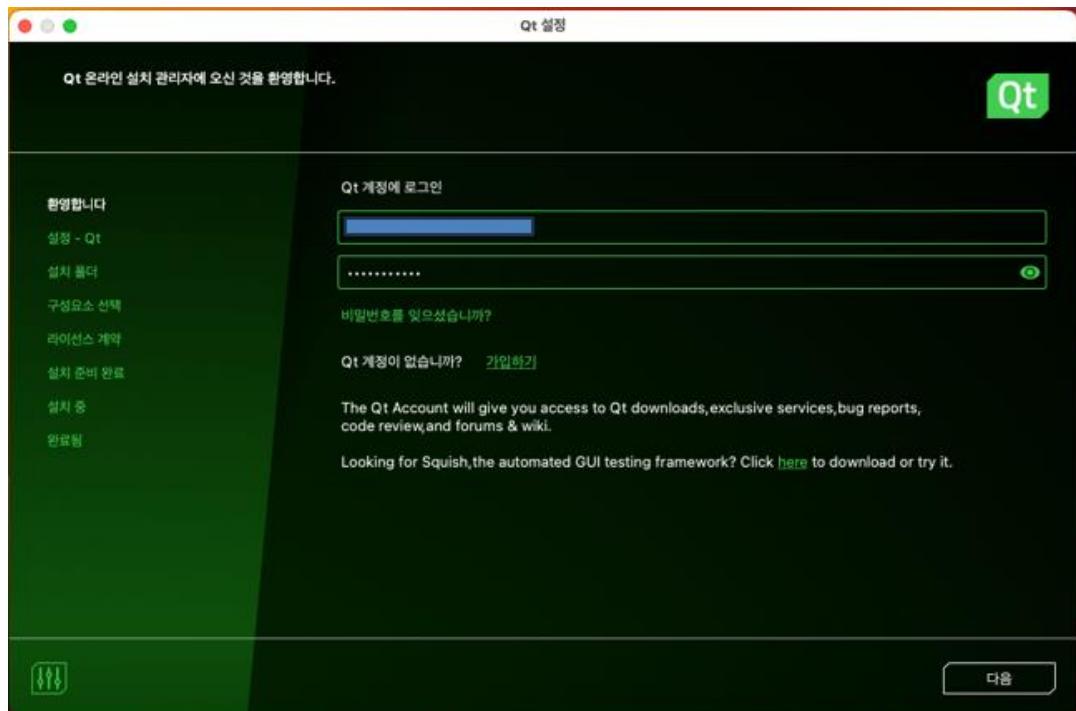
- Qt 설치

download.qt.io 사이트에서 Qt online install 파일을 다운로드 받는다. MacOS 용 파일은 확장자가 dmg이다.

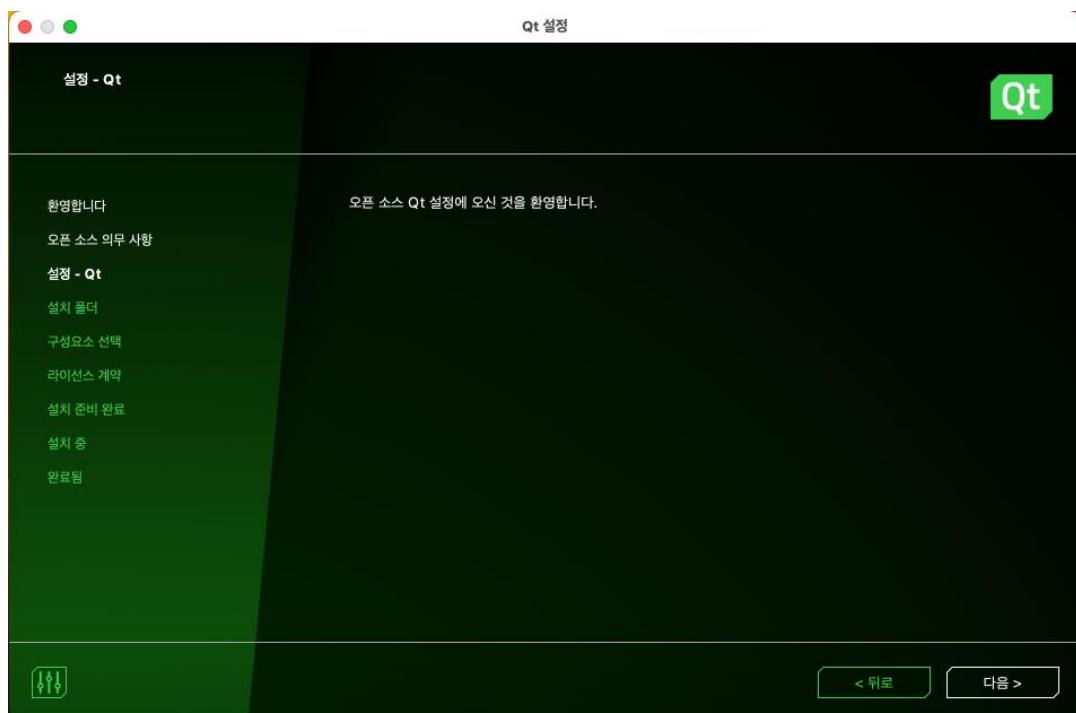
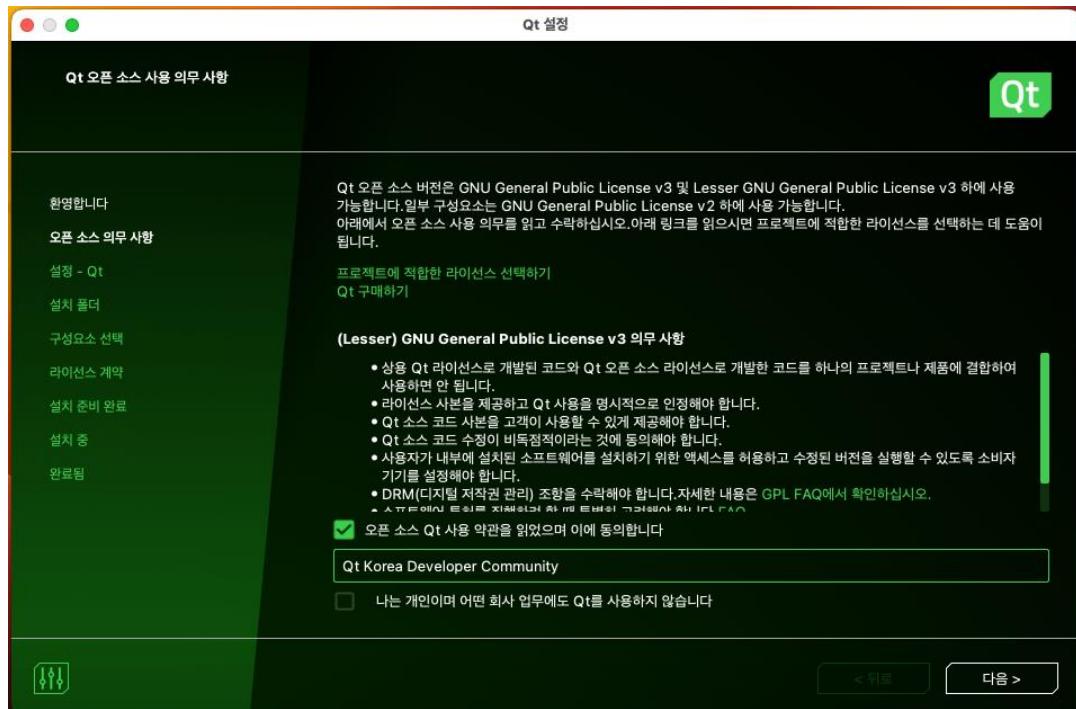
- Download URL: https://download.qt.io/official_releases/online_installers/
- Download file: qt-unified-mac-x64-online.dmg

Qt Downloads		Qt Home	Bug Tracker	Code Review	Planet Qt	Get Qt Extensions
Name	Last modified	Size	Metadata			
↑ Parent Directory	-	-	-			
qt-unified-windows-x64-online.exe	15-May-2024 10:07	47M	Details			
qt-unified-mac-x64-online.dmg	15-May-2024 10:07	20M	Details			
qt-unified-linux-x64-online.run	15-May-2024 10:07	66M	Details			
qt-unified-linux-arm64-online.run	15-May-2024 10:07	68M	Details			

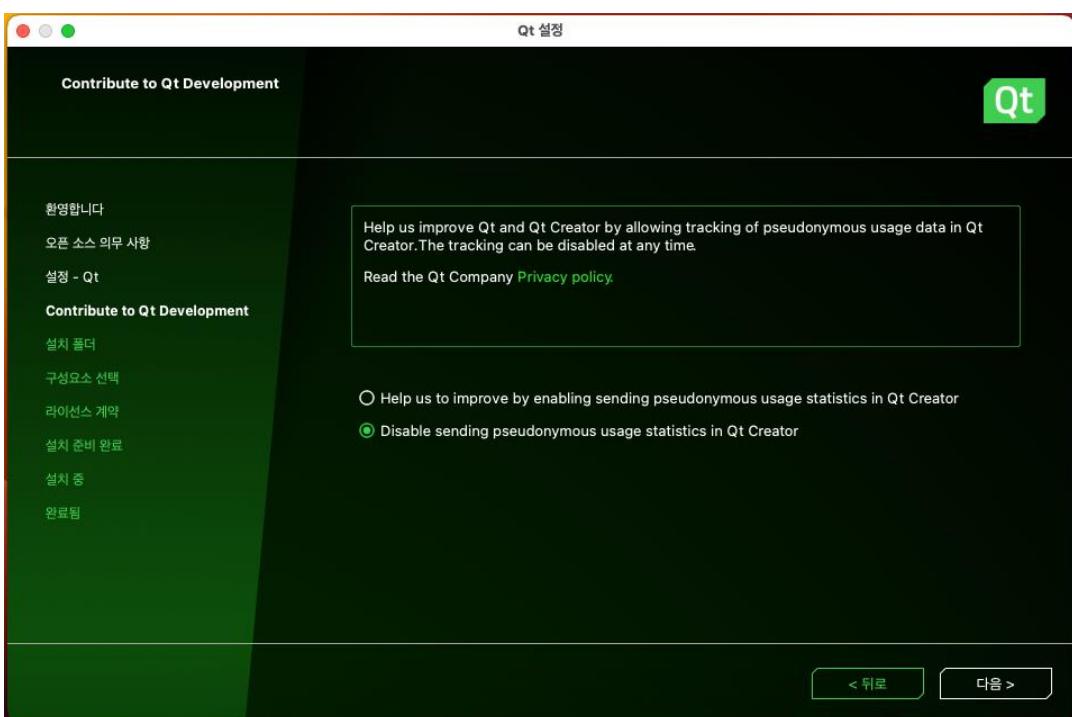
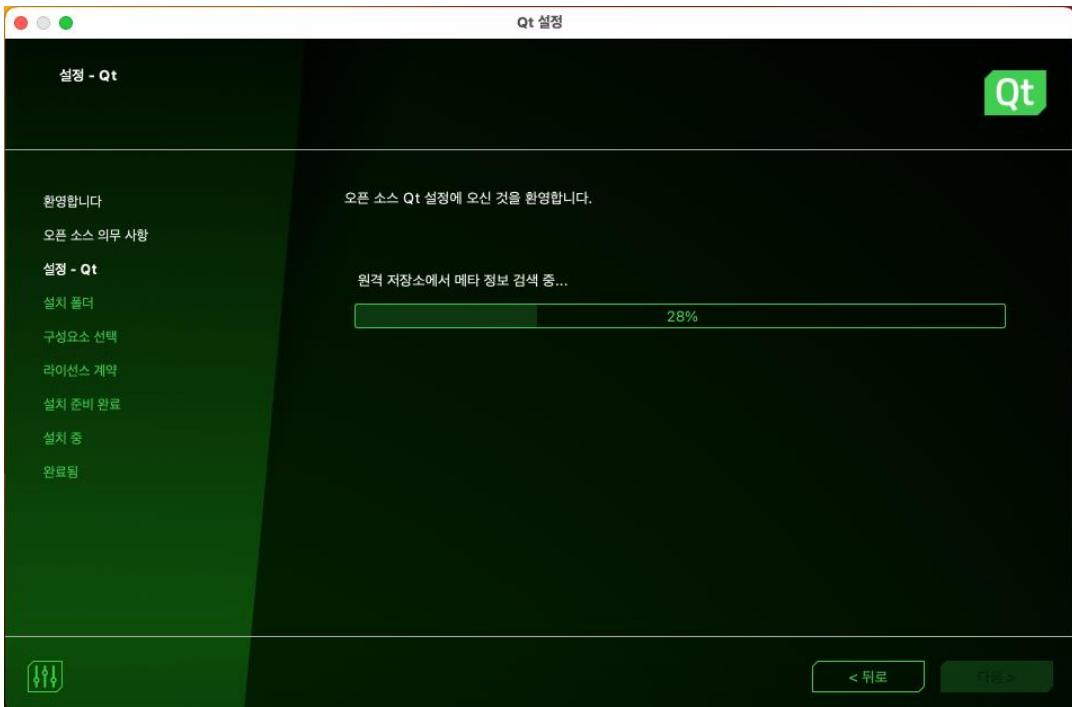
위에서 보는 것과 같이 Qt online install 파일을 다운로드 받은 후 실행한다.



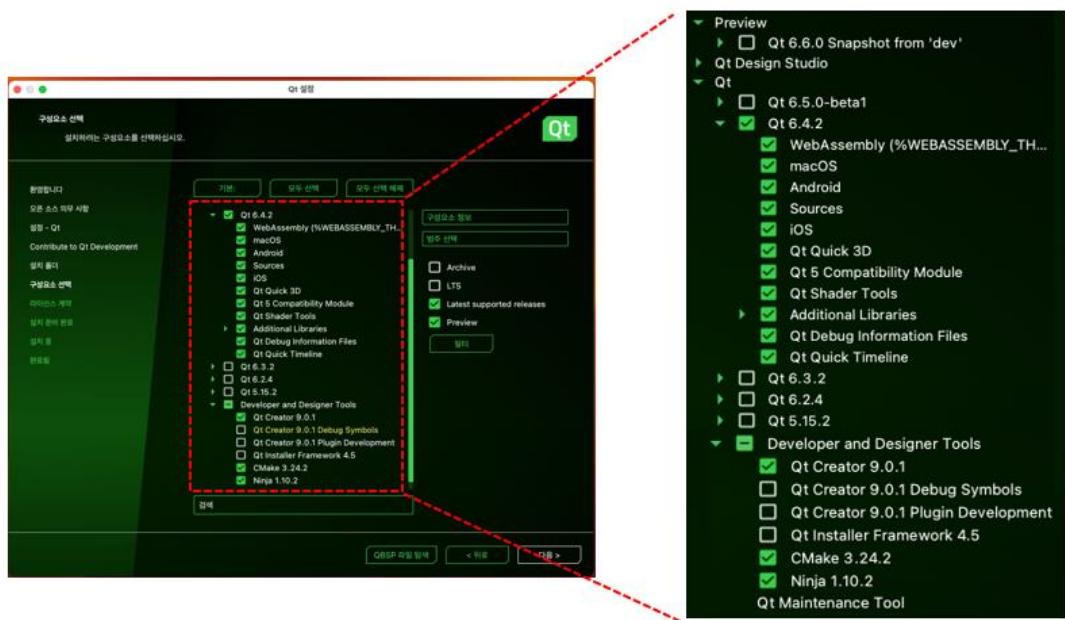
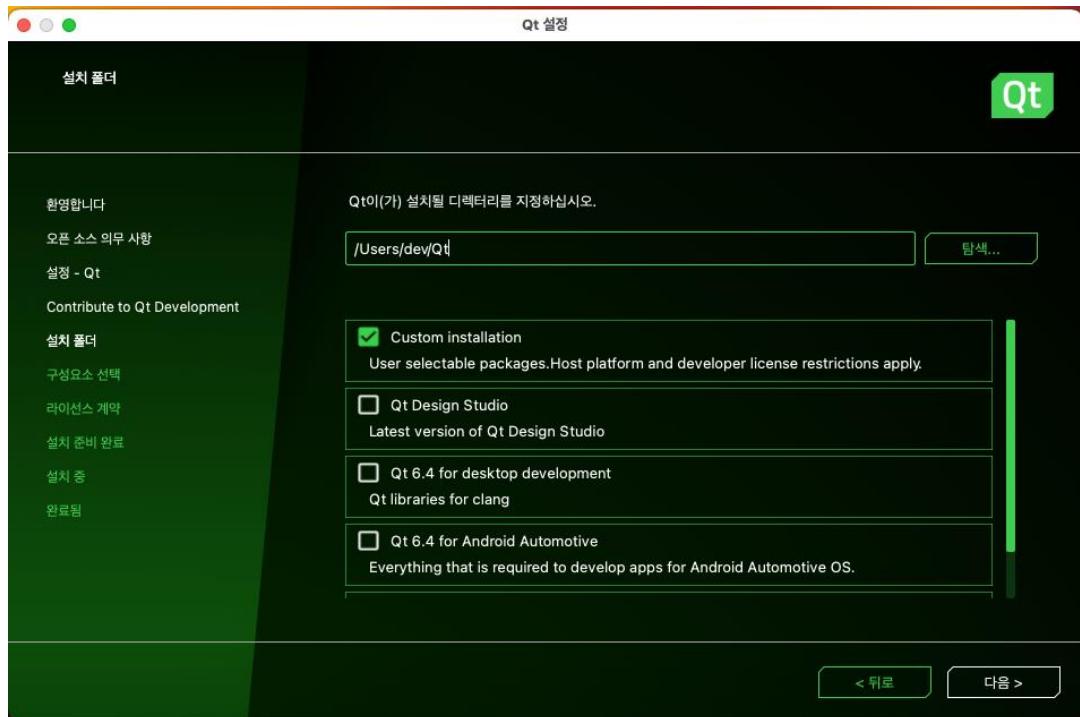
예수님은 당신을 사랑합니다.



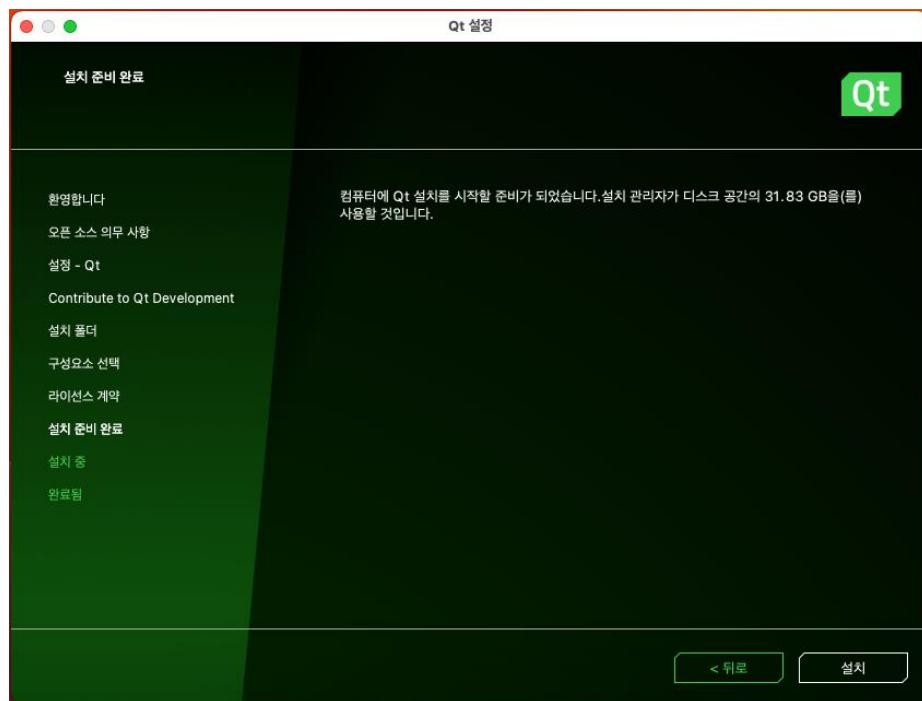
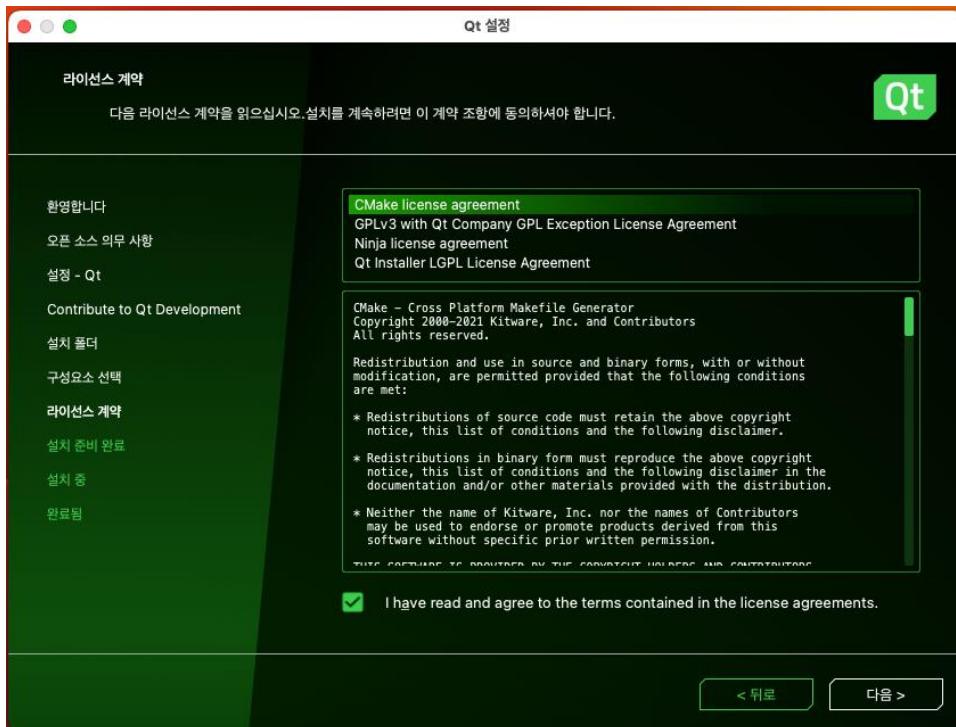
예수님은 당신을 사랑합니다.



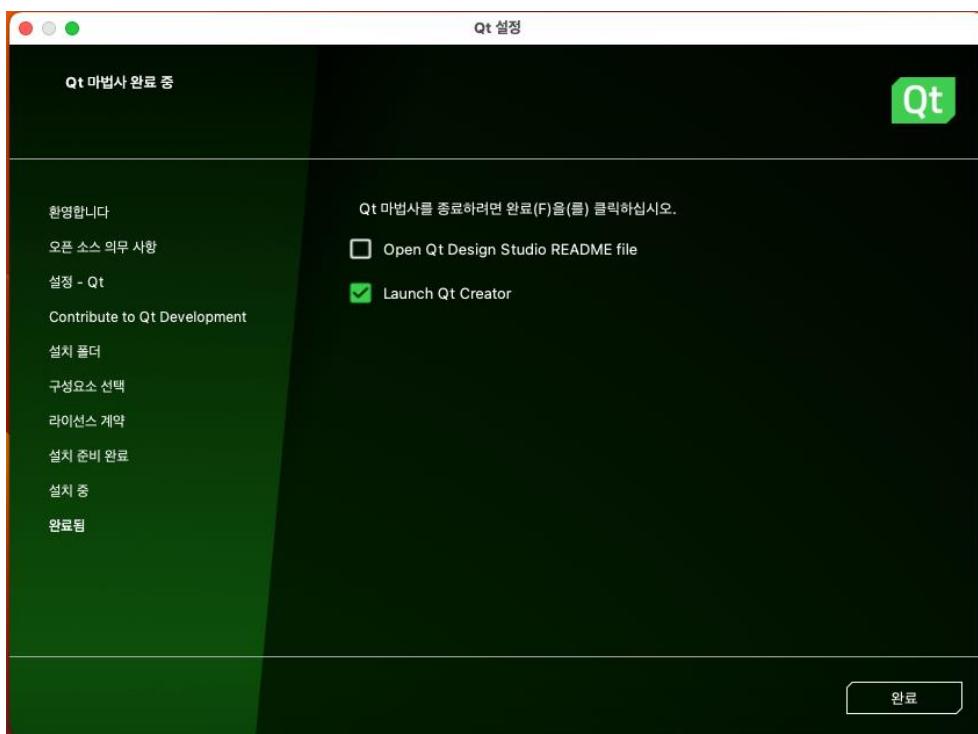
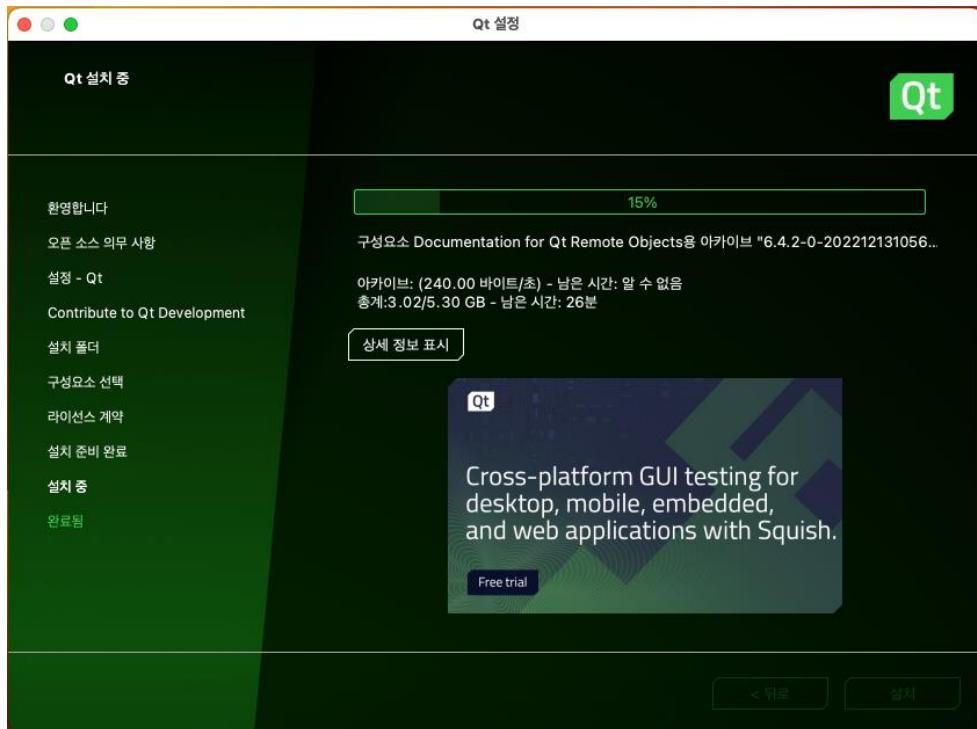
예수님은 당신을 사랑합니다.



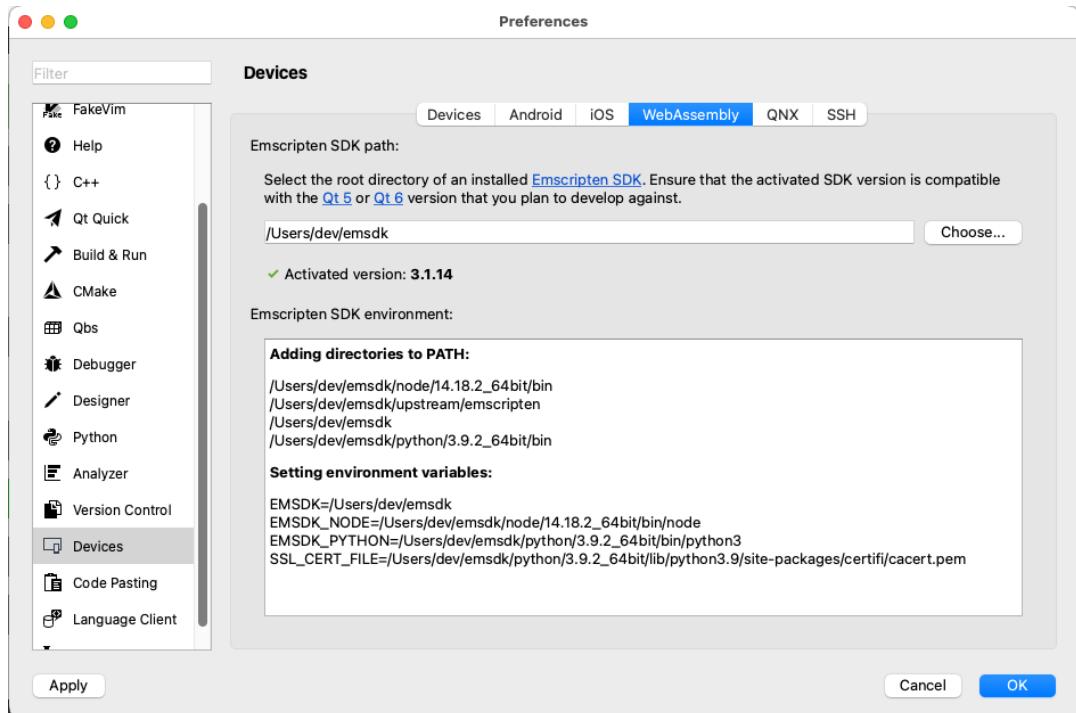
예수님은 당신을 사랑합니다.



예수님은 당신을 사랑합니다.



예수님은 당신을 사랑합니다.



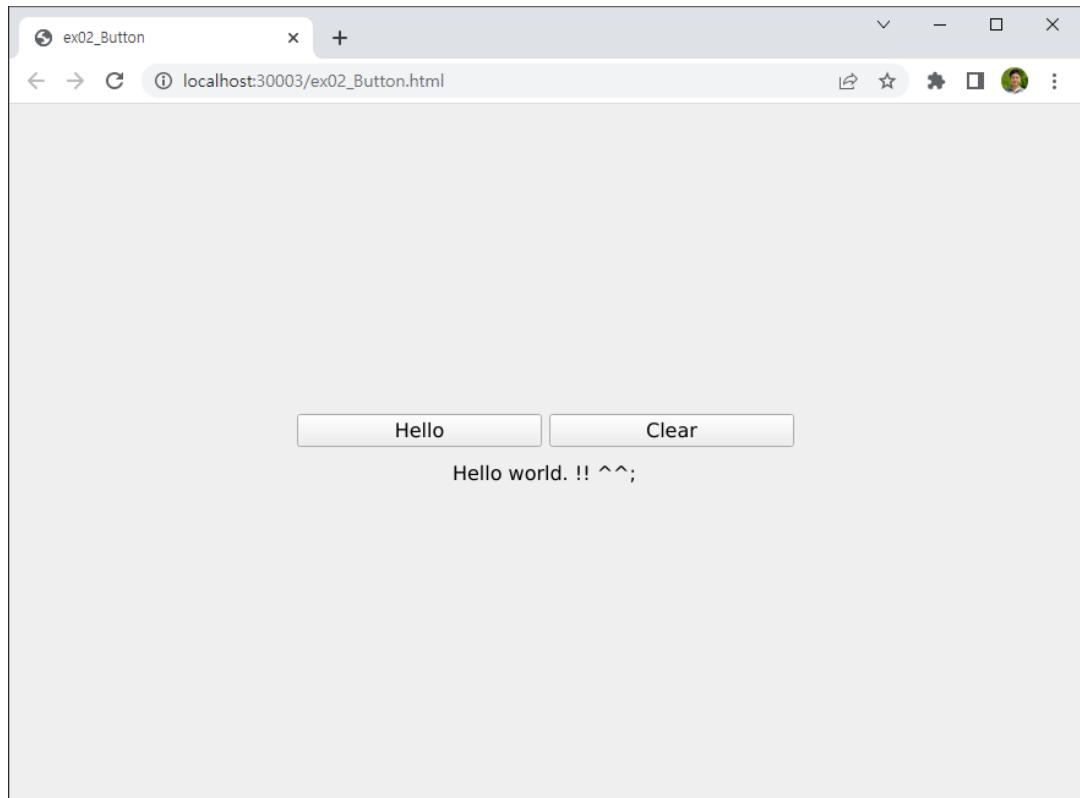
위의 그림에서 보는 것과 같이 왼쪽 탭에서 [Devices] 클릭한다. 그런 다음 오른쪽 탭에서 [WebAssembly] 탭을 선택한다. 그리고 위와 같이 버전을 선택한다. 여기서는 Qt 6.4 버전을 사용했다. 따라서 Emscripten 버전은 3.1.14를 사용했다.

만약 macOS에서 Qt 6.5 버전을 사용하려면 Emscripten 버전은 3.1.25 버전을 사용해야 한다.

5. Qt for WebAssembly 프로그래밍의 시작

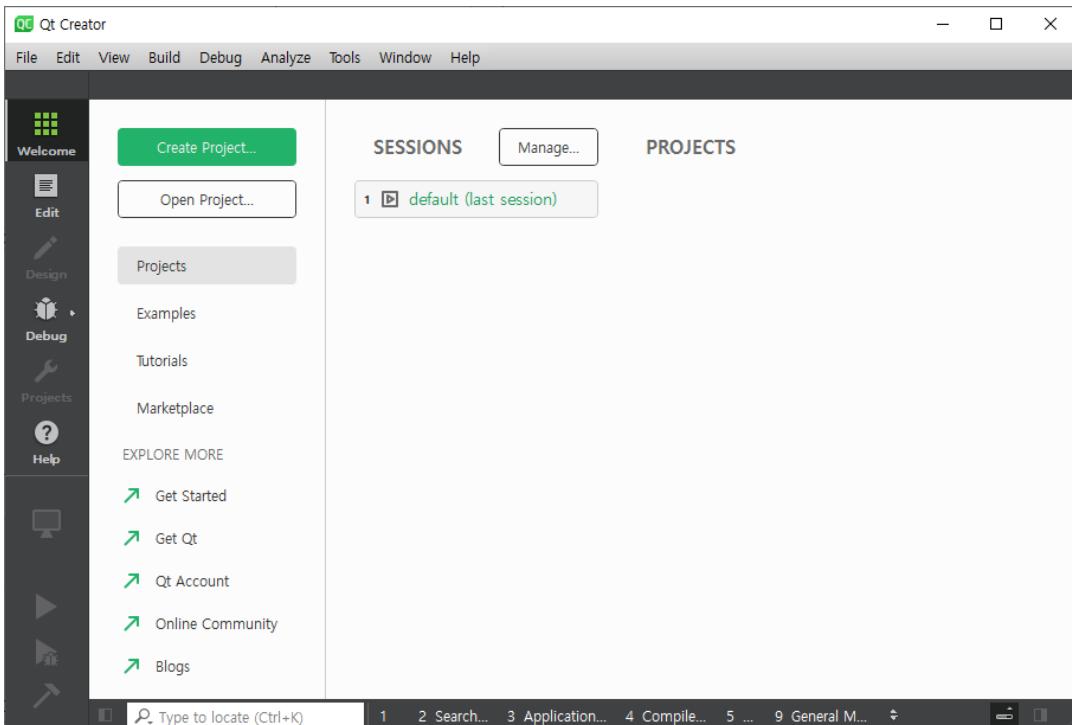
이번에는 우리가 Qt for WebAssembly를 이용해 간단한 Qt WebAssembly 어플리케이션을 구현해 보도록 하자.

구현할 어플리케이션은 아래 그림에서 보는 것과 같이 [Hello] 버튼을 클릭하면 "Hello world. !! ^^;" 메시지를 버튼 아래에 출력한다. 그리고 [Clear] 버튼을 클릭하면 버튼 아래의 메시지를 삭제하는 간단한 기능을 구현해 보도록 하자.

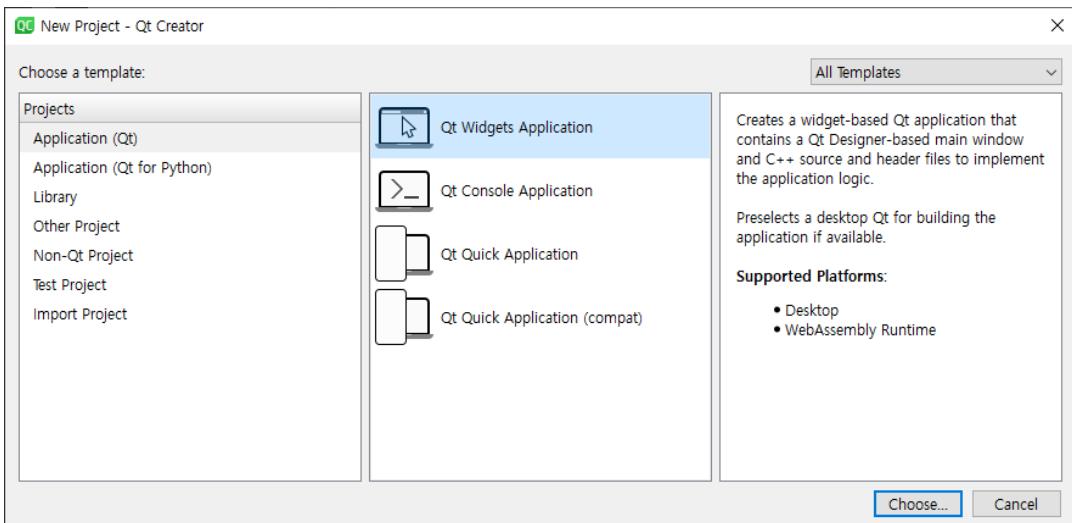


아래 그림에서 보는 것과 같이 Qt Creator를 실행한다.

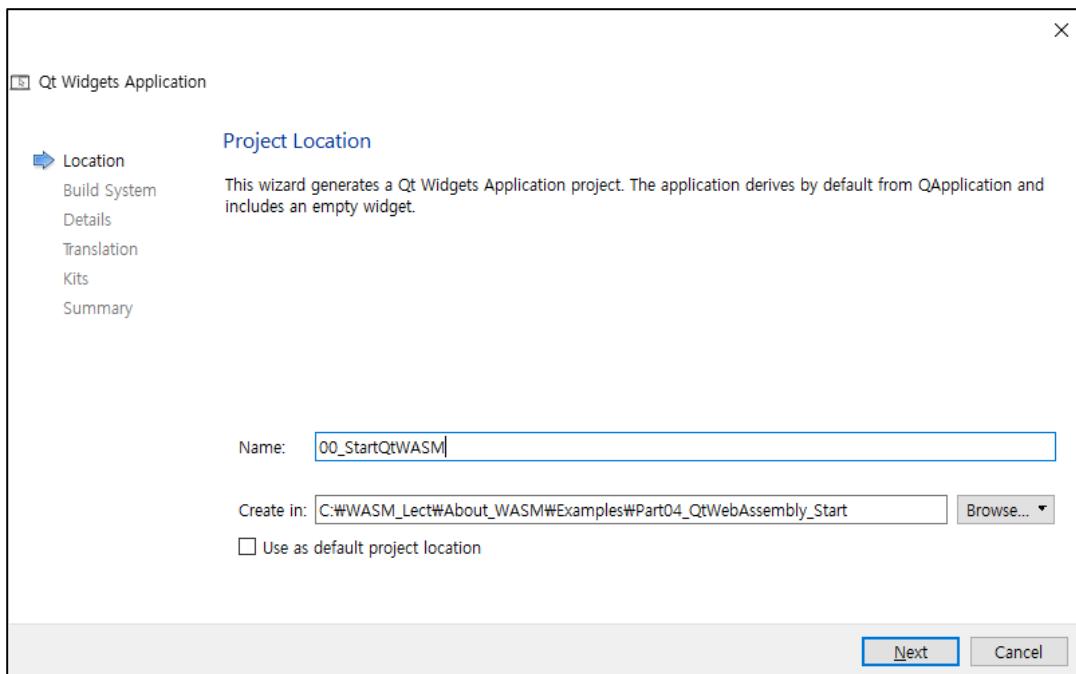
예수님은 당신을 사랑합니다.



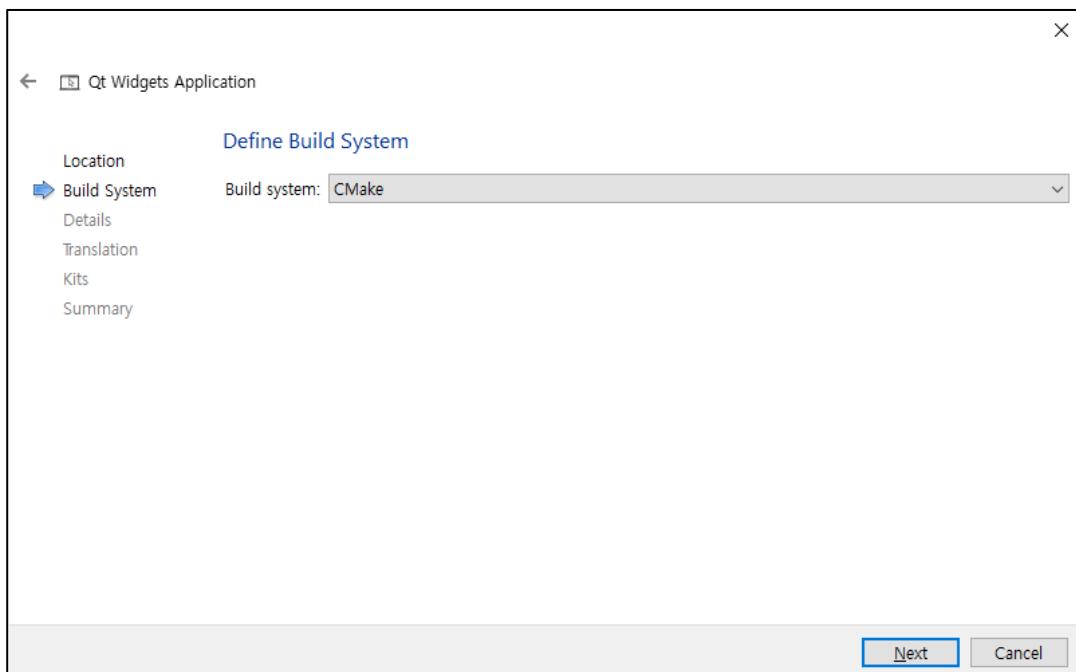
프로젝트를 생성하는 메뉴를 클릭한다.



위의 그림에서 보는 것과 같이 [Application (Qt)] 항목을 선택한다. 그리고 중간 탭에서 [Qt Widgets Application]을 선택한다.

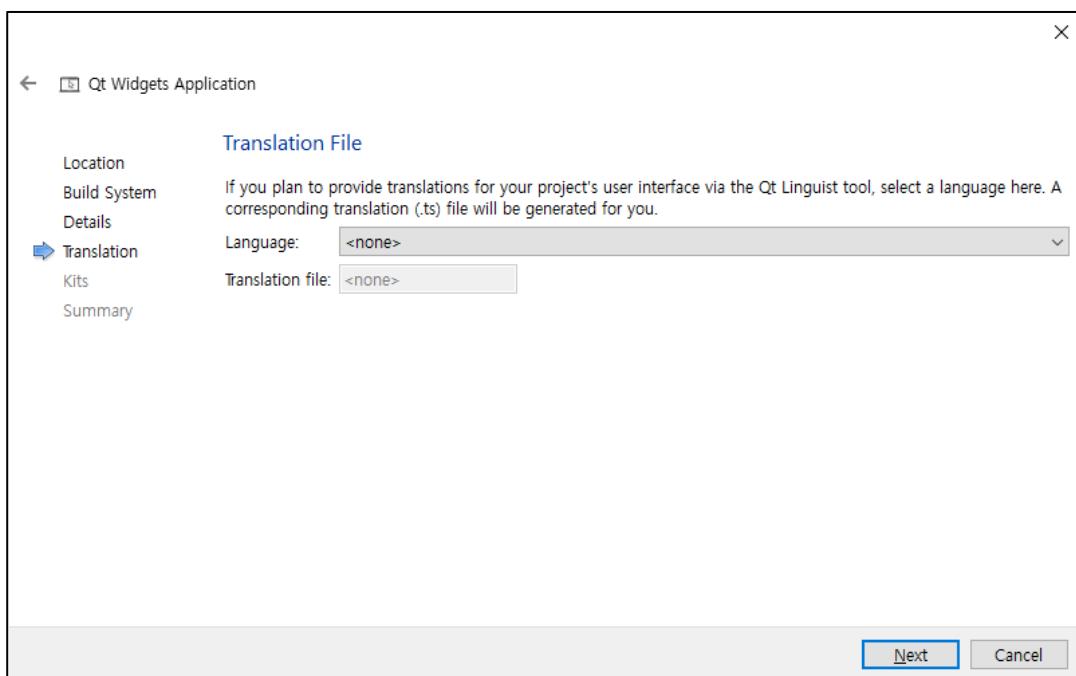
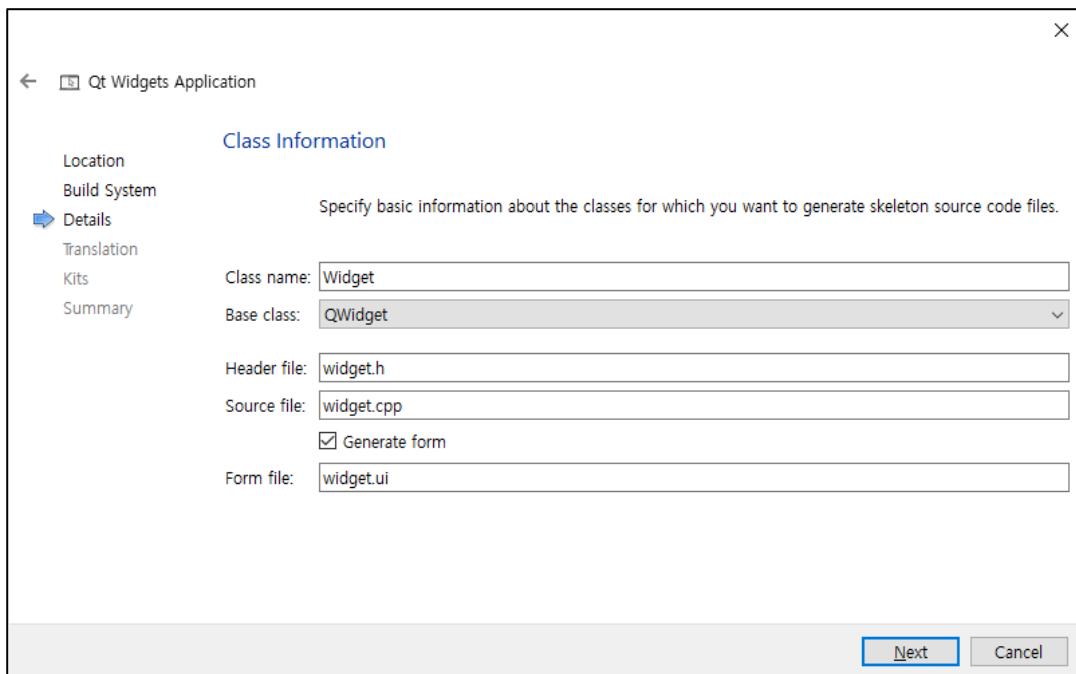


Buil System에서 CMake를 선택한다.

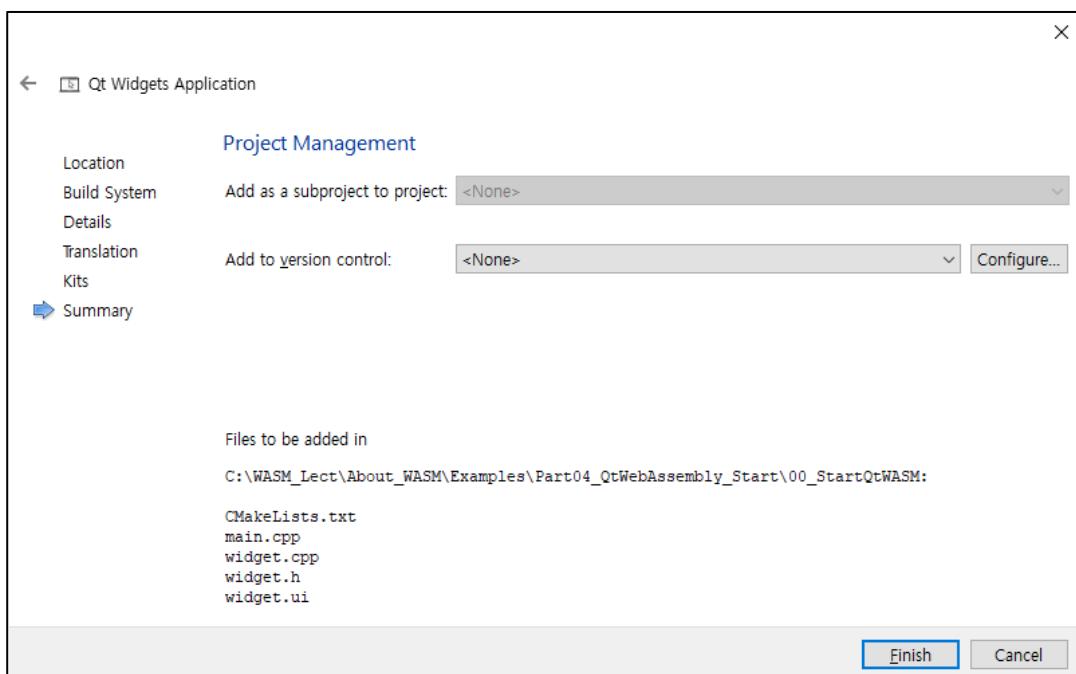
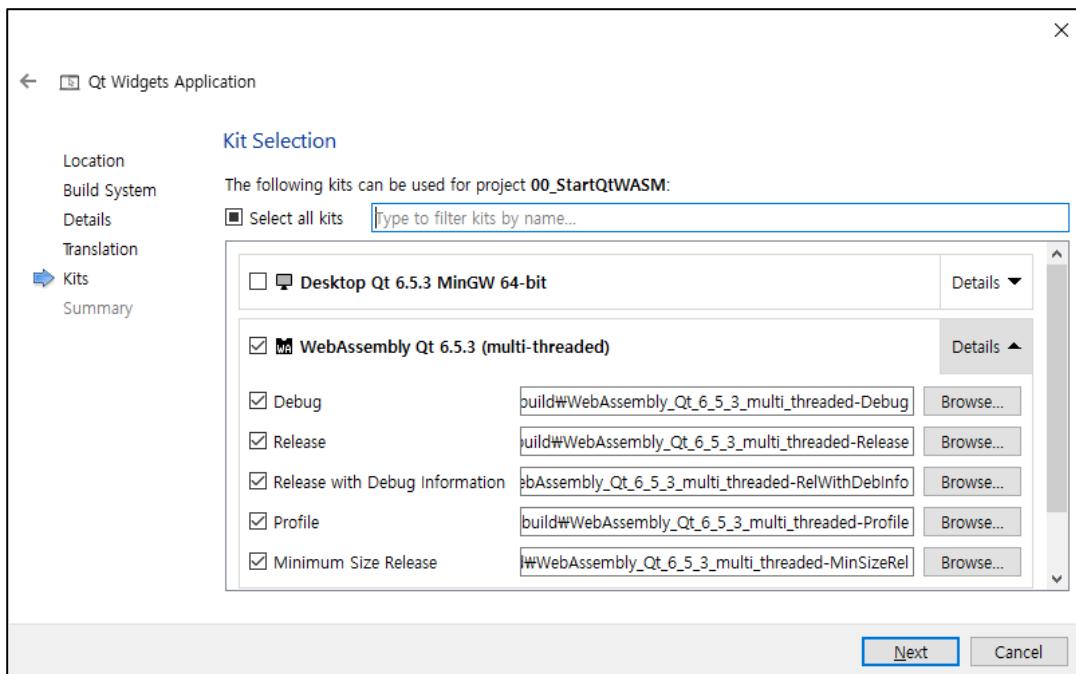


[Class Information] 다이얼로그에서 [Generate form] 체크박스를 체크한다.

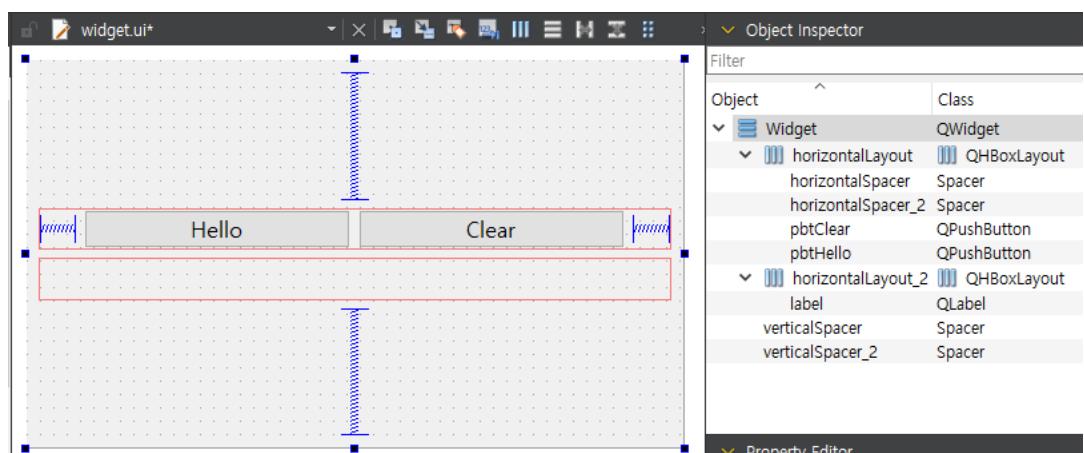
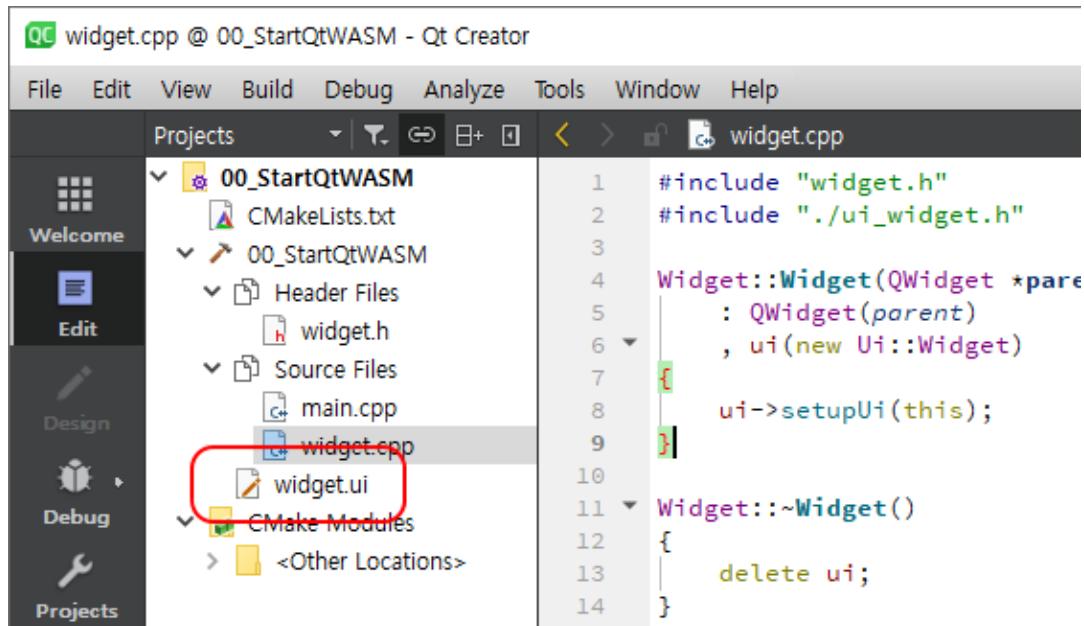
예수님은 당신을 사랑합니다.



[Kit Selection] 다이얼로그에서 [WebAssembly Qt 6.5.3 MinGW 64-bit] 항목을 선택한다.



프로젝트가 생성이 완료되면 [widget.ui] 파일을 더블 클릭한다. 그러면 Qt Designer 화면이 로딩될 것이다.



위와 같이 GUI의 Widget들을 배치하고 저장한다. 그리고 widget.h 헤더파일을 아래와 같이 작성한다.

```

#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>

QT_BEGIN_NAMESPACE
namespace Ui {

```

예수님은 당신을 사랑합니다.

```
class Widget;  
}  
QT_END_NAMESPACE  
  
class Widget : public QWidget  
{  
    Q_OBJECT  
  
public:  
    Widget(QWidget *parent = nullptr);  
    ~Widget();  
  
private:  
    Ui::Widget *ui;  
  
private slots:  
    void slot_helloBtn();  
    void slot_clearBtn();  
  
};  
#endif // WIDGET_H
```

다음으로 widget.cpp 소스코드를 아래와 같이 작성한다.

```
#include "widget.h"  
#include "./ui_widget.h"  
  
Widget::Widget(QWidget *parent)  
    : QWidget(parent)  
    , ui(new Ui::Widget)  
{  
    ui->setupUi(this);  
    connect(ui->pbtHello, SIGNAL(pressed()), this, SLOT(slot_helloBtn()));  
    connect(ui->pbtClear, SIGNAL(pressed()), this, SLOT(slot_clearBtn()));
```

예수님은 당신을 사랑합니다.

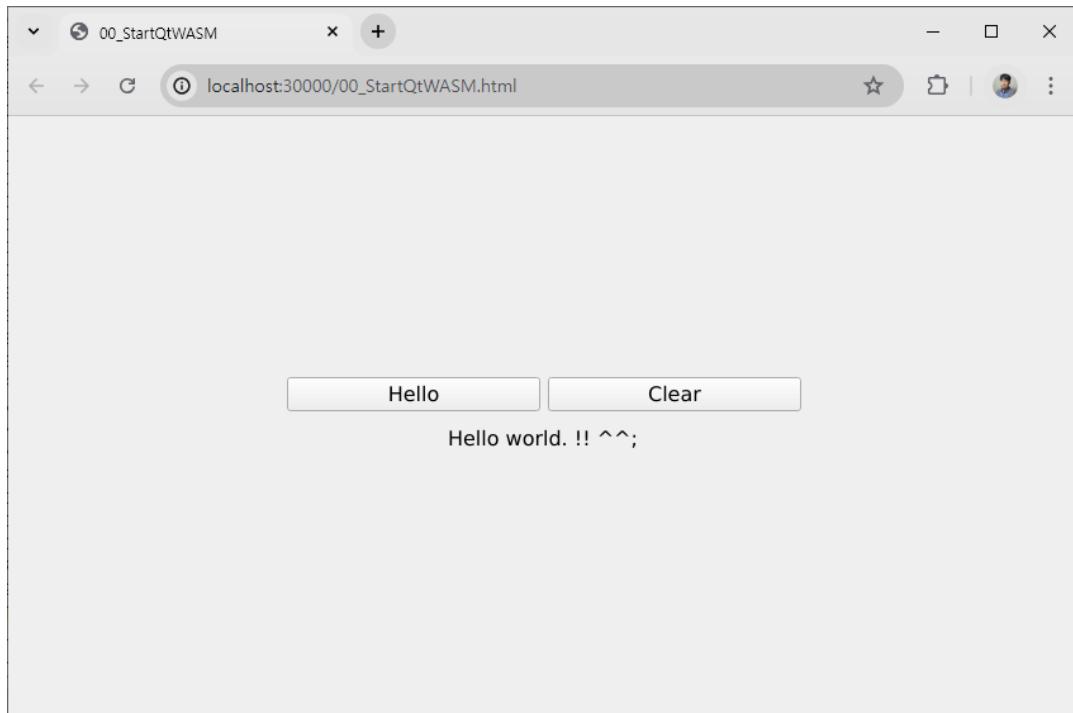
```
}

Widget::~Widget()
{
    delete ui;
}

void Widget::slot_helloBtn()
{
    ui->label->setText("Hello world. !! ^^;");
}

void Widget::slot_clearBtn()
{
    ui->label->setText("");
}
```

다음으로, 프로젝트를 빌드한다. 그런 다음에 실행해 보도록 하자.



예수님은 당신을 사랑합니다.

6. Signal and Slot

Qt는 이벤트를 처리하기 위한 메커니즘으로 시그널(Signal)과 슬롯(Slot)을 사용한다. 예로 어떤 버튼이 클릭했다는 행위는 Qt에서 시그널(Signal)이라고 한다. 그리고 시그널이 발생하면 호출하는 함수를 슬롯(Slot) 함수라고 한다. 시그널이라는 이벤트가 발생하면 시그널과 연결된 슬롯 함수가 호출된다.

시그널이란 어떠한 상황에 발생하는 이벤트이다. Qt의 모든 이벤트는 Signal과 Slot이라는 메커니즘을 사용한다.

예를 들어 Qt로 채팅 프로그램에서 A라는 사용자가 B라는 사용자에게 메시지를 보낸다고 가정해보자 B의 입장에서 A에게로부터 메시지를 받은 행위는 시그널(Signal)이라고 정의할 수 있다.

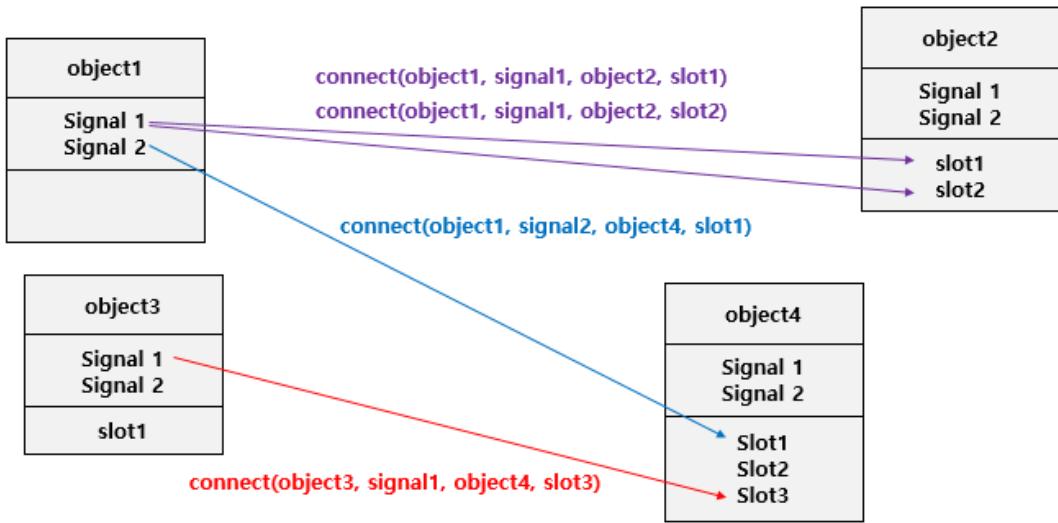
그리고 메시지를 받은 시그널과 연결된 슬롯(Slot) 함수를 호출한다. Qt는 모든 이벤트 처리를 시그널과 슬롯을 사용한다. GUI 와 네트워크 이외에도 Qt에서 제공하는 모든 모듈(API)에서 시그널과 슬롯이라는 이벤트 메커니즘을 사용한다. 시그널과 슬롯은 프로그램 소스코드를 단순화시켜주기 때문에 개발기간을 단축시킬 수 있으며 복잡한 프로그램 구조를 단순화할 수 있다.

네트워크 채팅 프로그램을 Qt의 시그널 슬롯을 사용하지 않고 채팅 프로그램을 개발한다고 가정해보자. 여러 개의 Thread 구조의 프로그램을 개발해야 한다. 예를 들어 특정 사용자가 보내는 메시지를 처리하는 쓰레드, 귓속말 처리 쓰레드, 새로운 사용자가 등록되면 알려주는 쓰레드 등 여러 개의 쓰레드를 사용해야 하기 때문에 프로그램이 복잡해질 수 있다.

하지만 Qt에서 시그널과 슬롯을 사용하면 쓰레드를 사용하지 않고도 간단하게 채팅 프로그램을 구현할 수 있다.

Qt에 제공하는 모든 GUI 위젯은 미리 정해진 다양한 시그널을 가지고 있다. 예를 들어 QPushButton의 click, double click, mouse over 등과 같이 다양한 시그널이 정의되어 있다.

시그널과 슬롯은 하나의 파이프라인과 같이 생각하면 된다. 하나의 시그널이 여러 개의 슬롯 함수를 호출할 수 있다. 또한 여러 개의 시그널이 하나의 슬롯을 호출할 수 있다.



시그널과 슬롯 함수를 연결하기 위한 함수는 QObject 클래스의 connect() 함수를 이용해 Signal 과 Slot을 연결할 수 있다. connect() 멤버 함수의 첫 번째 인자는 이벤트가 발생한 오브젝트(클래스의 인스턴스), 두 번째 인자는 오브젝트의 시그널(이벤트)을 입력한다.

예를 들어 A라는 버튼이 있으면 A라는 버튼의 오브젝트 명이 첫 번째 인자이고 두 번째 인자는 A버튼의 클릭 또는 더블클릭이 시그널이 될 수 있다.

따라서 클릭 이벤트를 두번째 인자로 명시한다. 세 번째 인자는 시그널과 호출할 슬롯 함수 있는 오브젝트의 이름을 명시한다. 네 번째 인자는 발생한 시그널 발생 시 호출할 슬롯 함수를 명시한다.

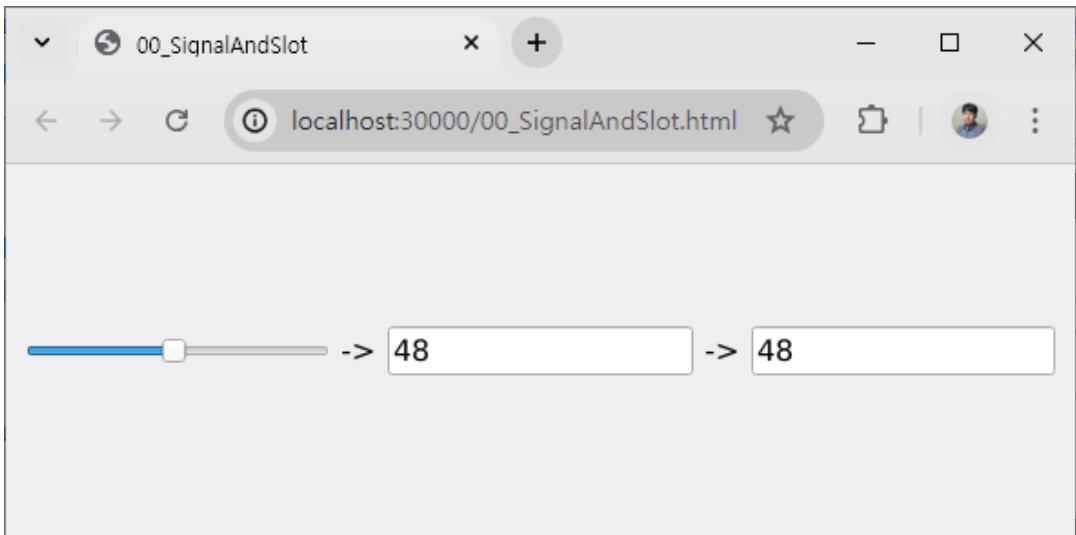
지금까지 Signal 과 Slot에 대한 개념에 대해 살펴보았다. 이번에는 직접 Signal 과 Slot 을 이용해 예제 프로그램 작성해 보도록 하자.

● Signal 과 Slot 예제

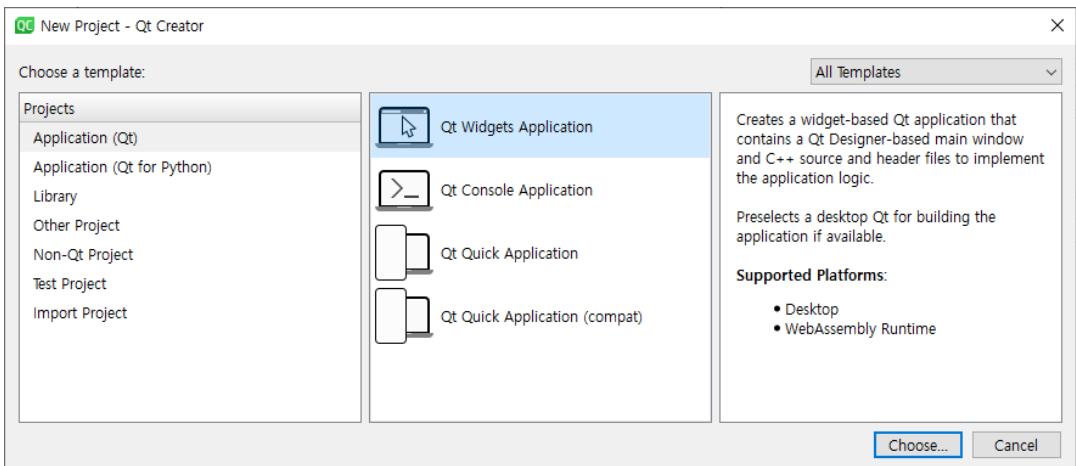
아래 그림에서 보는 것과 같이 첫 번째 QSlider위젯에서 Signal이 발생하면, 이 Signal과 연결된 Slot 함수를 호출한다.

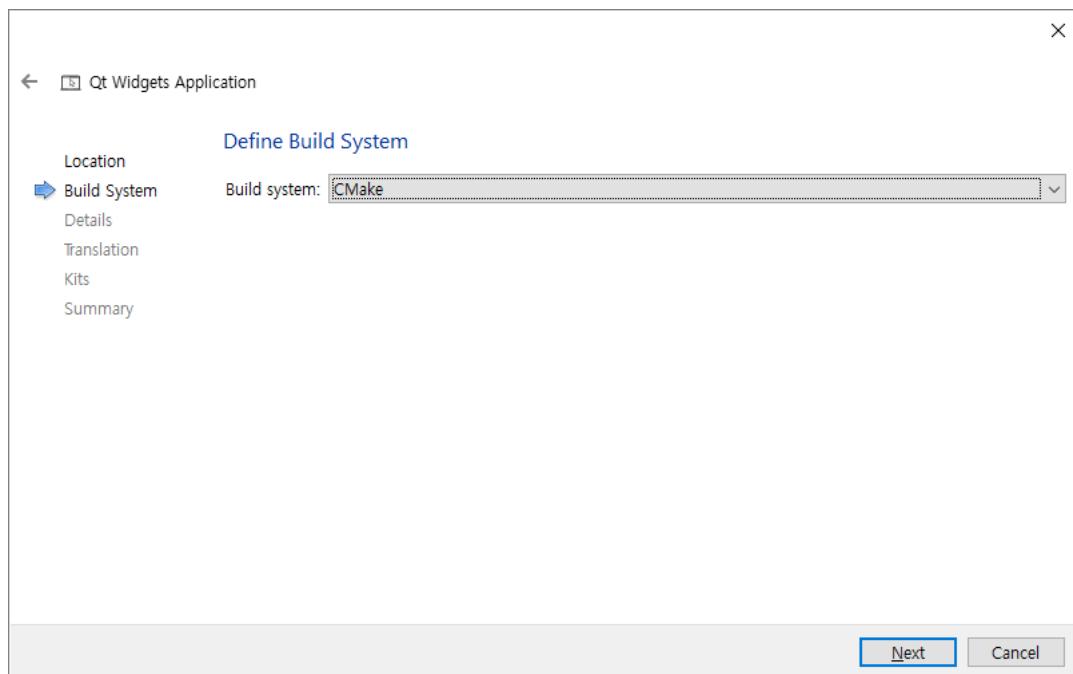
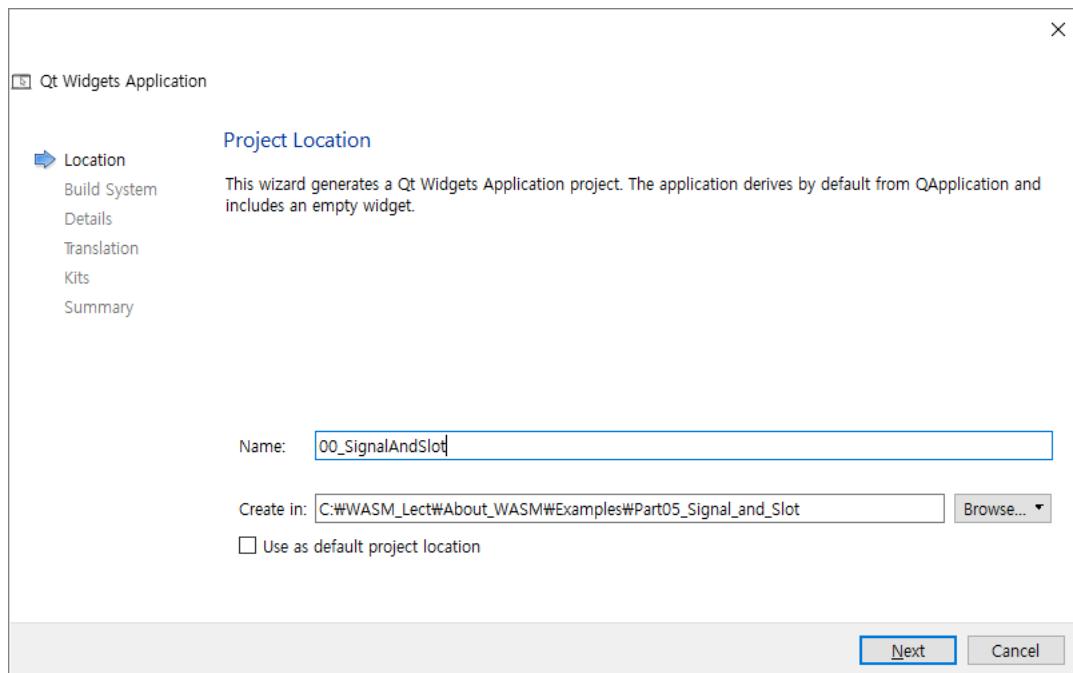
이 Slot 함수는 QSlider의 값을 두 번째 QLineEdit 위젯에 표시한다. 그리고 이 Slot 함수에서 선언한 sig_textChanged() 시그널을 실행한다. 그러면 이 Signal 과 연결된 Slot()함수를 실행하고 그 Slot 함수에서는 세 번째 QLineEdit 위젯에 값을 출력한다.

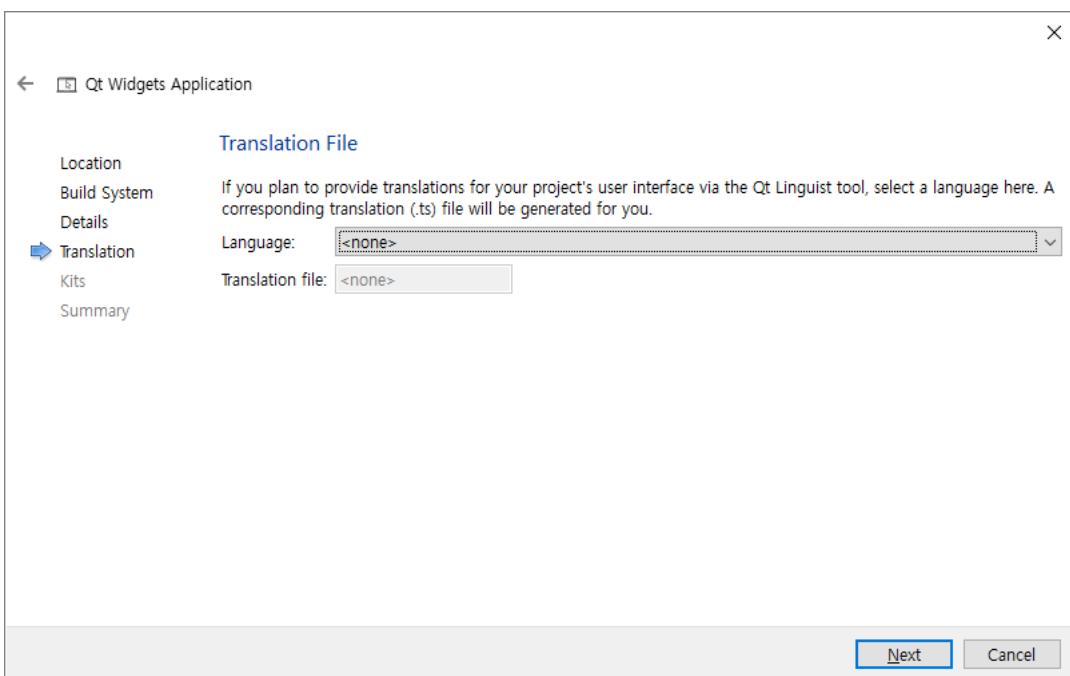
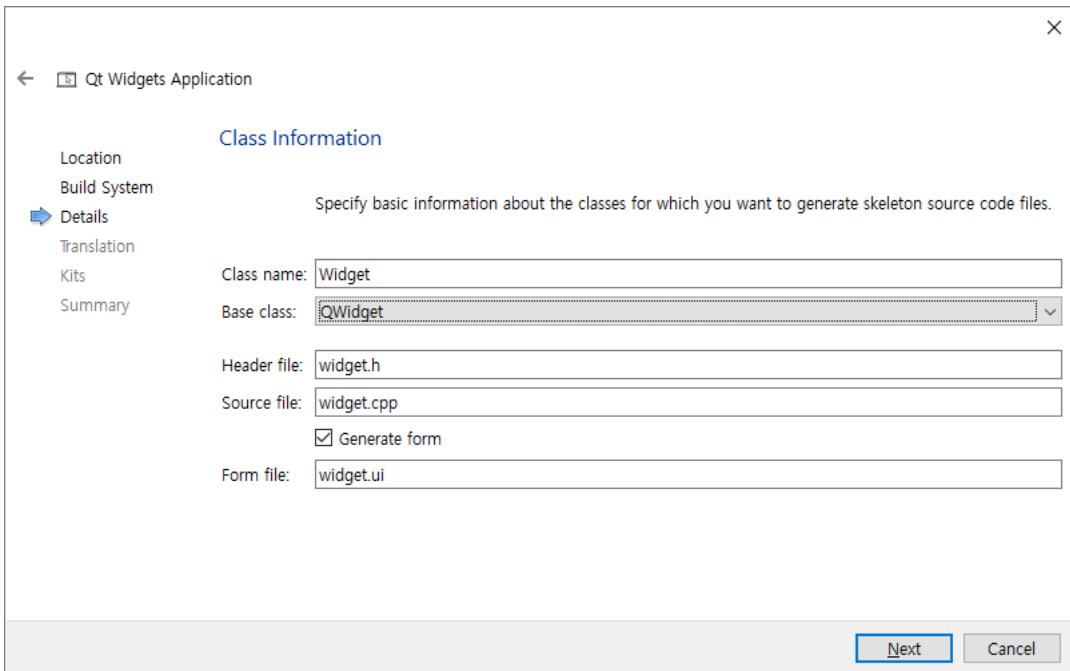
예수님은 당신을 사랑합니다.

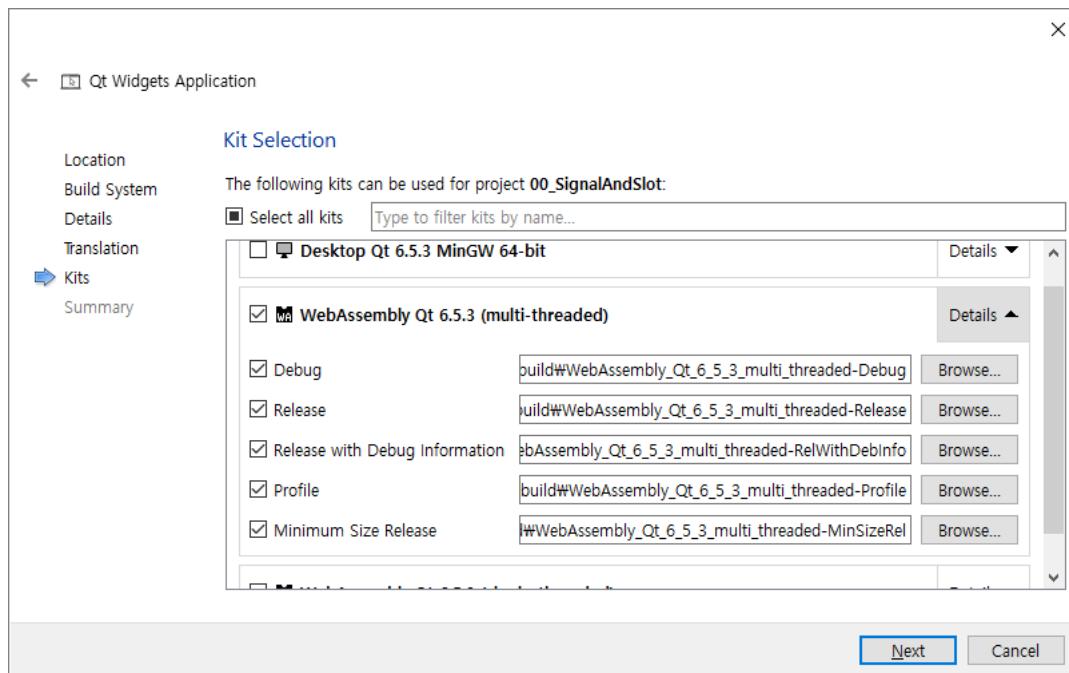


Qt Creator에서 아래 그림에서 보는 것과 같이 프로젝트를 생성한다.

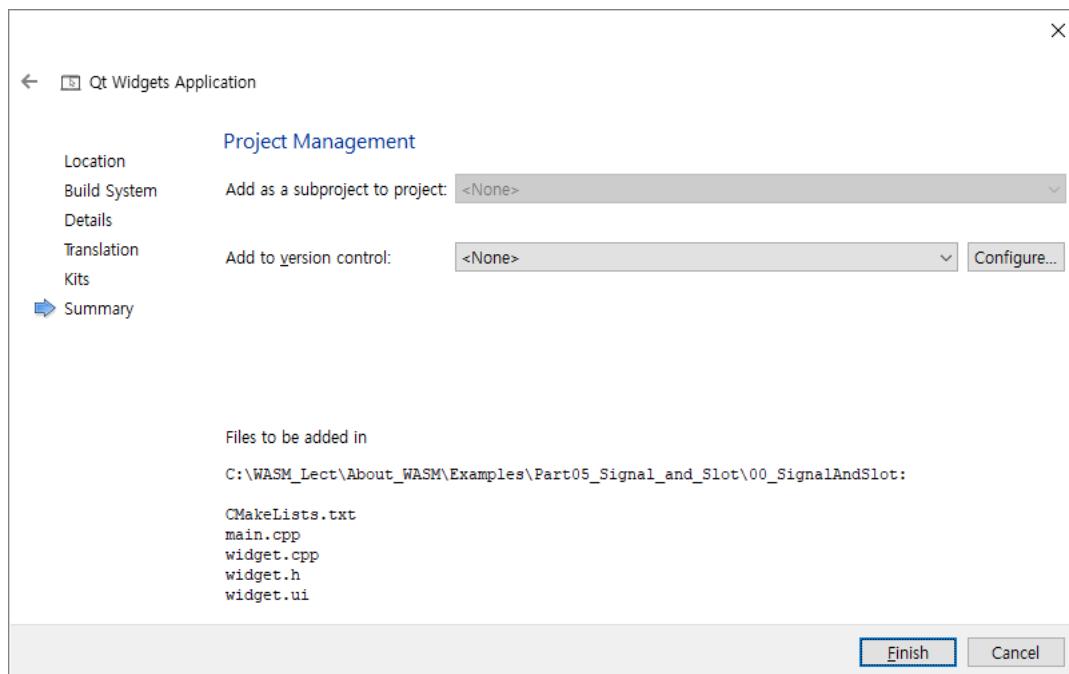






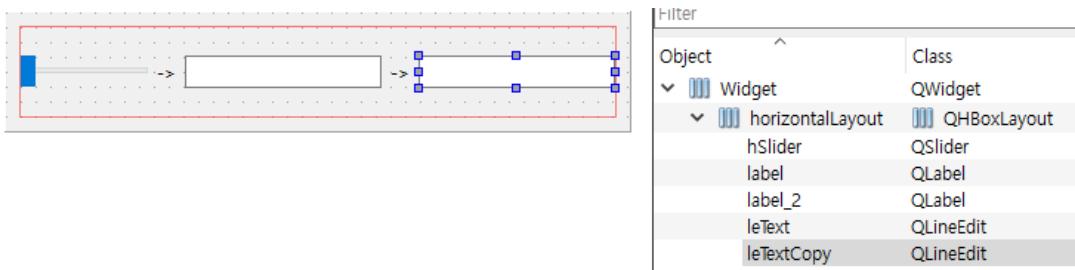


위의 그림에서 보는 것과 같이 [WebAssembly Qt 6.x.x (multi-threaded)] 항목을 선택한다.



위와 같이 프로젝트를 생성하고 Qt Creator에서 widget.ui를 더블 클릭한다. 그러면 Designer에서 아래 그림과 같이 GUI들을 배치한다.

예수님은 당신을 사랑합니다.



위와 같이 작성한 다음에 widget.h 헤더파일을 아래와 같이 작성한다.

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>

QT_BEGIN_NAMESPACE
namespace Ui {
class Widget;
}
QT_END_NAMESPACE

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();

private:
    Ui::Widget *ui;

signals:
    void sig_textChanged(QString str);
}
```

private slots:

```
void slotValueChanged(int val);
void slotTextChanged(QString str);

};

#endif // WIDGET_H
```

위의 그림에서 보는 것과 같이 `sig_textChanged()` 시그널을 추가한다. 이 시그널은 `slot_ValueChanged()` 함수에서 호출한다.

그리고 다음으로 `slot_ValueChanged()` 함수와 `slotTextChanged()` 함수를 추가한다.

QSlider의 값이 변경되면 `slot_ValueChanged(int val)` 함수가 호출된다. 그리고 변경한 QSlider의 값을 첫 번째 QLineEdit의 값으로 설정한다. 그런 다음 `sigTextChanged()` 시그널을 호출한다. 이 시그널 함수는 `slotTextChanged()` 함수와 연결되어 있다. 따라서 이 함수에서 첫 번째 QLineEdit의 값과 동일한 값이 두 번째 QLineEdit에 설정된다.

다음으로 widget.cpp 소스코드 파일을 아래와 같이 작성한다.

```
#include "widget.h"
#include "./ui_widget.h"

Widget::Widget(QWidget *parent)
    : QWidget(parent)
    , ui(new Ui::Widget)
{
    ui->setupUi(this);

    connect(ui->hSlider, SIGNAL(valueChanged(int)),
            this, SLOT(slot_ValueChanged()));

    connect(this, SIGNAL(sigTextChanged(QString)),
            this, SLOT(slotTextChanged(QString)));
}
```

예수님은 당신을 사랑합니다.

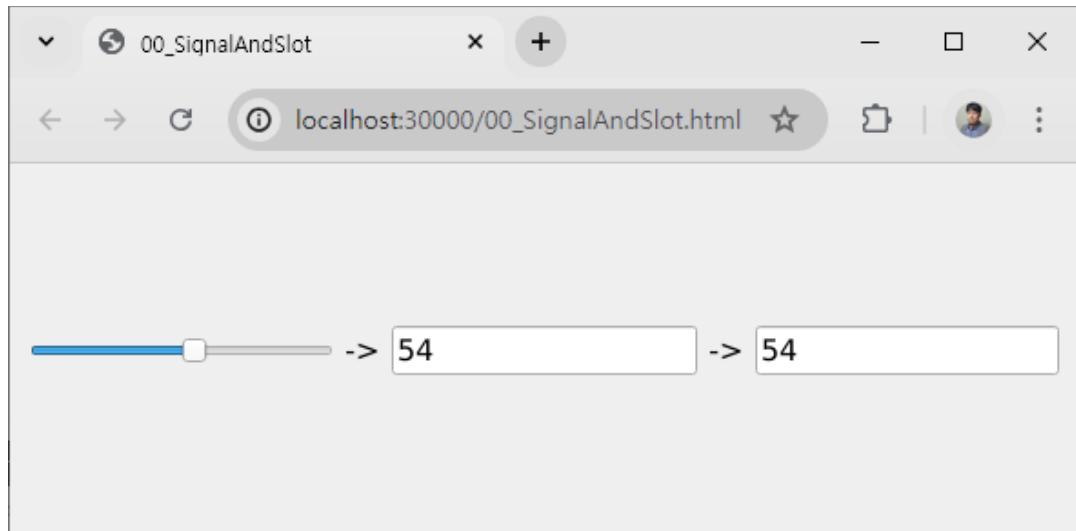
```
Widget::~Widget()
{
    delete ui;
}

void Widget::slotValueChanged(int value)
{
    QString str = QString("%1").arg(value);
    ui->leText->setText(str);

    emit sigTextChanged(str);
}

void Widget::slotTextChanged(QString str)
{
    ui->leTextCopy->setText(str);
}
```

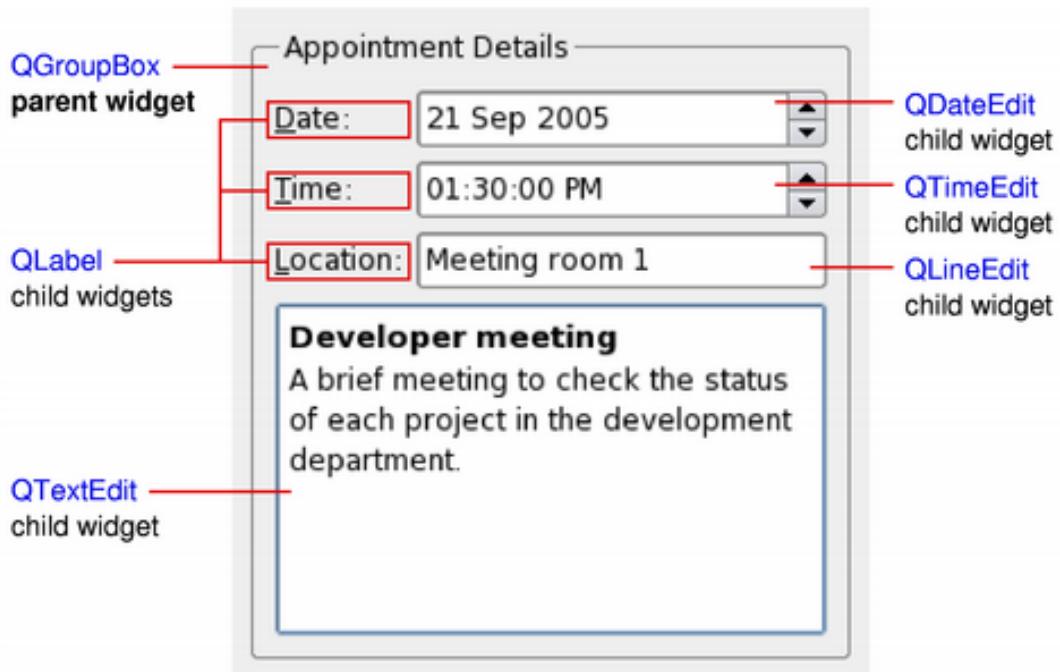
위와 같이 소스코드 작성은 완료한 다음 빌드 후 실행해 보도록 하자.



예수님은 당신을 사랑합니다.

7. Widget and Layouts

이번장에서는 Widget과 Layout에 대해서 알아보도록 하자. Widget이란 Button, ComboBox, Label 등을 Widget이라고 한다. 그리고 Layout이란 Widget들을 배치할 때 사용한다. Layout을 사용하면 Window의 크기가 변하면 Layout안에 있는 Widget들의 크기가 동적으로 변하게 된다.



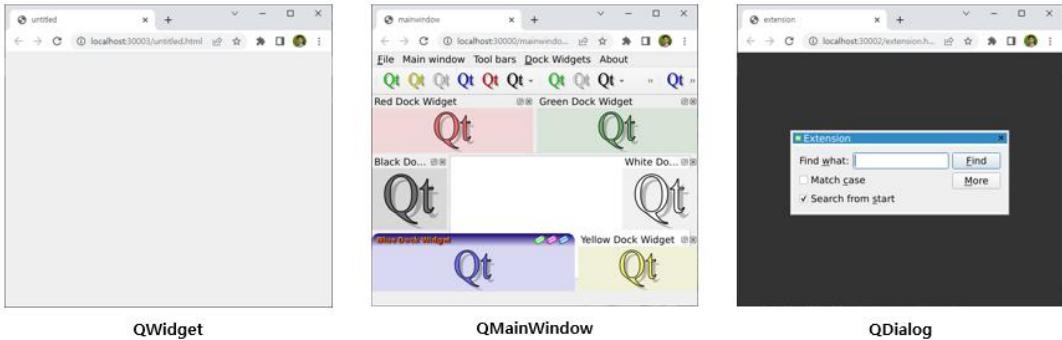
- QWidget, QMainWindow 그리고 QDialog

QWidget이란 마우스, 키보드 등과 같은 GUI상에서 발생하는 이벤트 처리, 2D 그래픽과 같은 도형, 선 등을 Drawing 할 수 있으며 이미지를 랜더링 할 수 있다. Qt에서 제공하는 모든 Widget들은 QWidget을 기반으로 구현되어졌다.

QMainWindow는 Menu Bar, Tool Bar, Center Widget 영역, Status Bar영역등으로 나누어져 있으며 Desktop PC에서 사용하는 GUI 어플리케이션에서 주로 사용된다.

QMainWindow도 QWidget을 상속받아 구현되었다. 또한 QDialog도 QWidget 기반으로 구현되었다.

예수님은 당신을 사랑합니다.

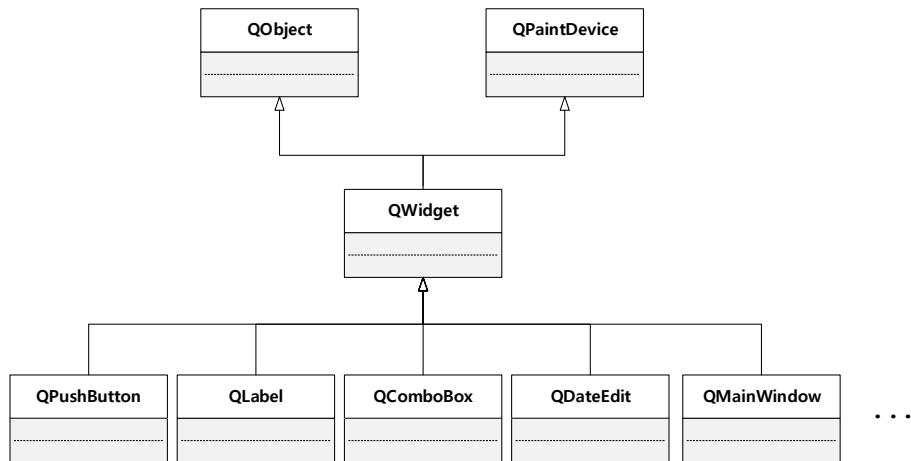


QWidget

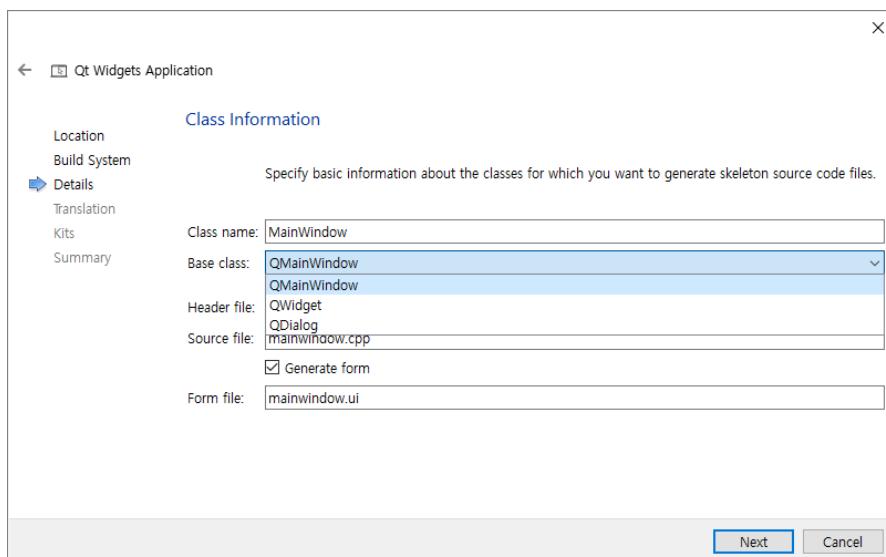
QMainWindow

QDialog

QWidget은 QObject와 QPaintDevice를 상속받는다. QObject는 Signal 과 Slot 등과 같은 기능이 구현되어 있고 QPaintDevice는 2D 그라픽관련 기능이 구현되어 있다.

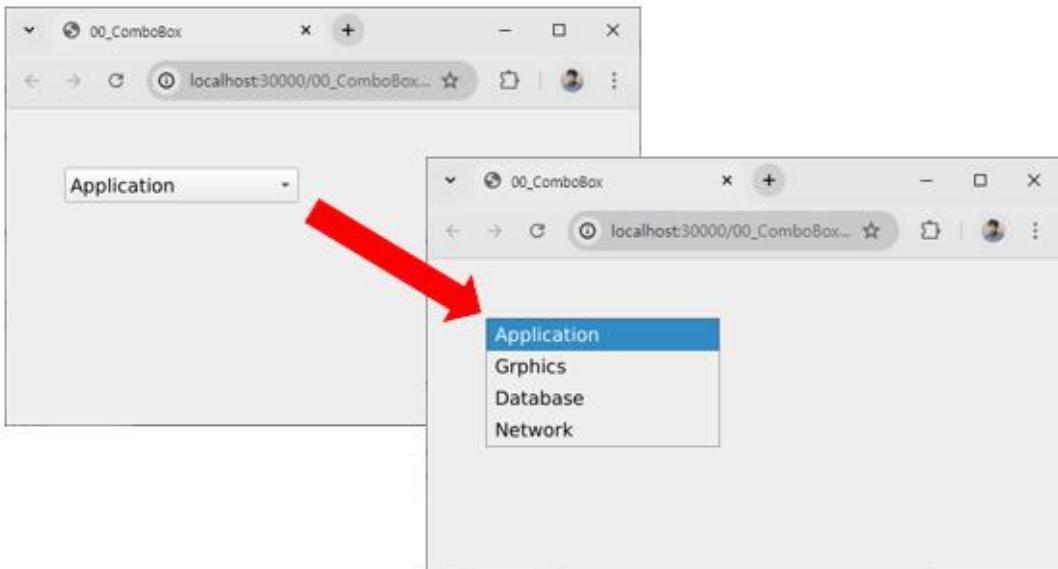


Qt WebAssembly는 최초 프로젝트 생성 시, Qt Widget 기반 어플리케이션을 선택했을 때, 다음과 같은 디아얼로그에서 생성하려는 윈도우가 어떤 위젯인지 생성할 수 있다.



- QComboBox

QComboBox는 아래 그림에서 보는 것과 같이 항목 중 하나를 선택할 수 있는 GUI 인터페이스를 아래 그림에서 보는 것과 같이 제공한다.



```
#include "widget.h"
#include <QComboBox>

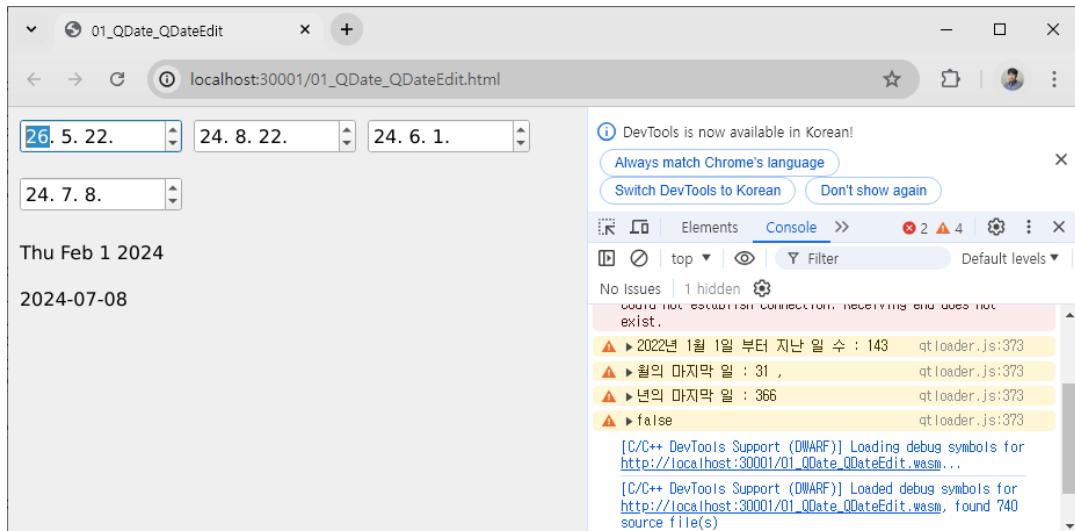
Widget::Widget(QWidget *parent)
    : QWidget(parent)
{
    QComboBox *combo = new QComboBox(this);
    combo->setGeometry(50, 50, 200, 30);

    combo->addItem("Application");
    combo->addItem("Graphics");
    combo->addItem("Database");
    combo->addItem("Network");
}
```

```
Widget::~Widget()
{
}
```

- QDate 와 QDateEdit

QDate는 날짜를 핸들링 하기위한 기능을 제공한다. 그리고 QDate 클래스 오브젝트를 GUI상에 표시하기 위해서 QDateEdit 위젯을 사용할 수 있다.



```
QDateEdit *dateEdit[4];
QLabel *lbl[6];

QDate dt1 = QDate(2024, 5, 22);
QDate dt2 = QDate::currentDate();

dateEdit[0] = new QDateEdit(dt1.addYears(2), this);
dateEdit[0]->setGeometry(10, 10, 140, 30);

dateEdit[1] = new QDateEdit(dt1.addMonths(3), this);
dateEdit[1]->setGeometry(160, 10, 140, 30);
```

```
dateEdit[2] = new QDateEdit(dt1.addDays(10), this);
dateEdit[2]->setGeometry(310, 10, 140, 30);

dateEdit[3] = new QDateEdit(dt2, this);
dateEdit[3]->setGeometry(10, 60, 140, 30);

qDebug("2022년 1월 1일부터 지난 일 수 : %d", dt1.dayOfYear());
qDebug("월의 마지막 일 : %d , ", dt1.daysInMonth());
qDebug("년의 마지막 일 : %d", dt1.daysInYear());

if(QDate::isValid(2024, 2, 41))
    qDebug("true");
else
    qDebug("false");

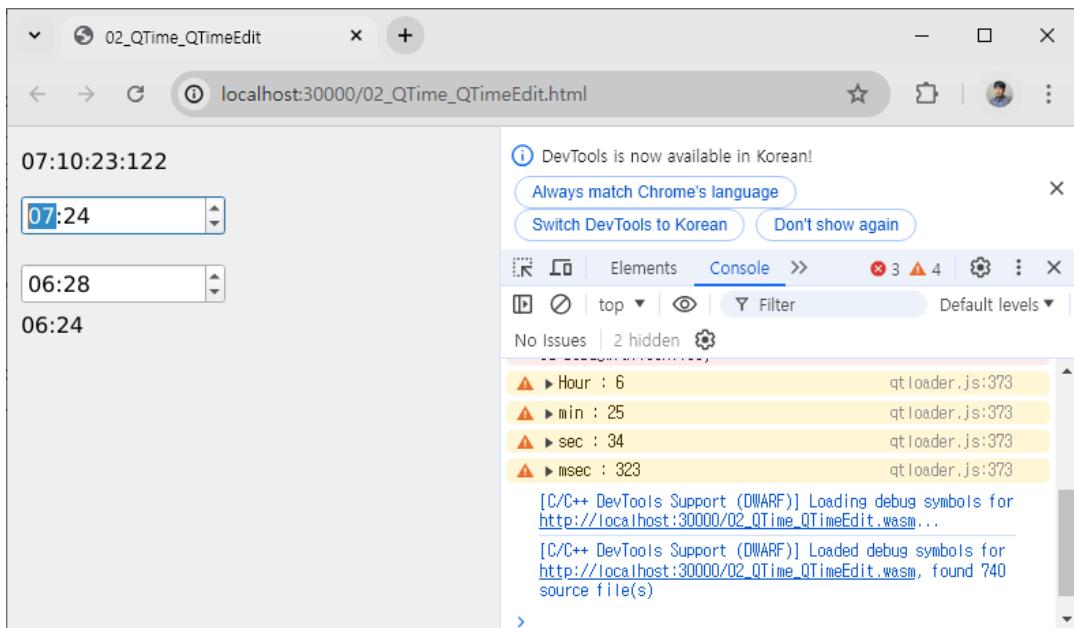
QDate dt3 = QDate(2024, 2, 1);
QDate dt4 = QDate::currentDate();

lbl[0] = new QLabel(dt3.toString(), this);
lbl[0]->setGeometry(10,110, 150, 30);
lbl[1] = new QLabel(dt4.toString("yyyy-MM-dd"), this);
lbl[1]->setGeometry(10,150, 150, 30);
```

- QTime 과 QTimeEdit

QTime은 시간을 핸들링 할 수 있는 기능을 제공한다. 그리고 QTimeEdit는 QTime을 이용해 GUI상에 표시하는 기능을 제공한다.

예수님은 당신을 사랑합니다.



```
QTime ti0 = QTime(7, 10, 23, 122);
```

```
QLabel *lbl_toString;
```

```
Lbl_toString = new QLabel(ti0.toString("hh:mm:ss:zzz"), this);
```

```
lbl_toString->setGeometry(10, 10, 150, 30);
```

```
// 시,분,초,밀리초(millisecond)
```

```
QTime ti1 = QTime(6, 24, 55, 432);
```

```
QTimeEdit *qte[2];
```

```
qte[0] = new QTimeEdit(ti1, this);
```

```
qte[0]->setDisplayFormat("hh:mm");
```

```
qte[0]->setGeometry(10, 50, 150, 30);
```

```
qte[1] = new QTimeEdit(ti1.addSecs(200), this);
```

```
qte[1]->setDisplayFormat("hh:mm");
```

```
qte[1]->setGeometry(10, 100, 150, 30);
```

```
QLabel *lbl_fromString = new QLabel(ti1.toString("hh:mm"), this);
```

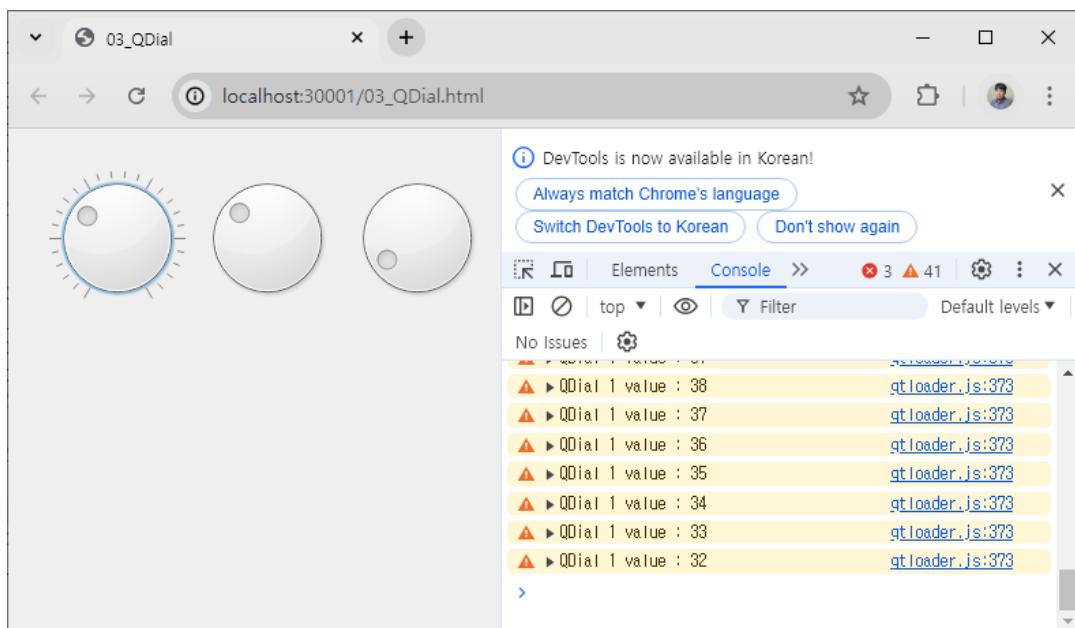
예수님은 당신을 사랑합니다.

```
lbl_fromString->setGeometry(10, 130, 150, 30);
QTime ti5 = QTime(6, 25, 34, 323);

qDebug("Hour : %d", ti5.hour());
qDebug("min : %d", ti5.minute());
qDebug("sec : %d", ti5.second());
qDebug("msec : %d", ti5.msec());
```

● QDial

QDial은 사용자에게 Dial과 같은 GUI인터페이스를 제공한다.



```
#include "widget.h"
Widget::Widget(QWidget *parent) : QWidget(parent)
{
    int xpos = 30;
    for(int i = 0 ; i < 3 ; i++, xpos += 110)
    {
        m_dial[i] = new QDial(this);
```

```
m_dial[i]->setRange(0, 100);
m_dial[i]->setGeometry(xpos, 30, 100, 100);
}

m_dial[0]->setNotchesVisible(true);
connect(m_dial[0], SIGNAL(valueChanged(int)),
        this,           SLOT(changedData()));

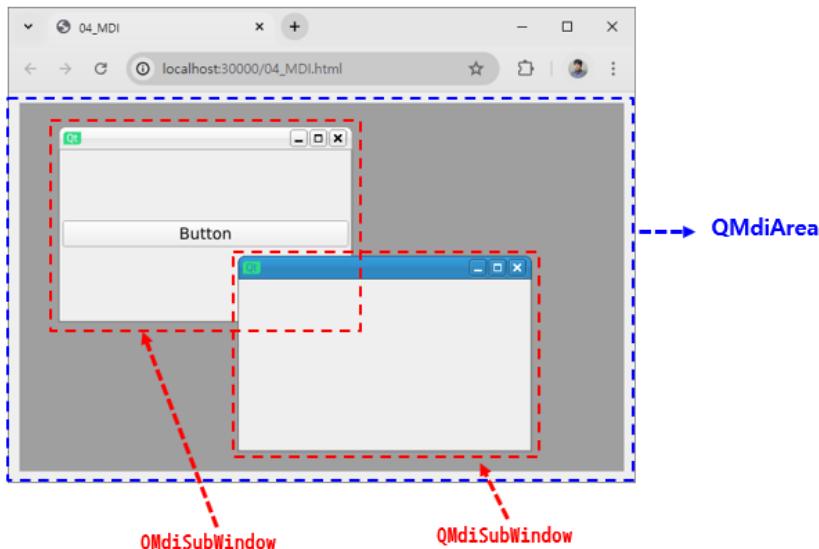
}

void Widget::changedData()
{
    qDebug("QDial 1 value : %d", m_dial[0]->value());
}

Widget::~Widget()
{
```

- QMdiArea

QMdiArea는 윈도우창 안에 여러 개의 윈도우를 생성할 수 있다. 따라서 윈도우창 안에 Sub 윈도우를 생성 및 사용할 수 있다.



```
#include "widget.h"
```

```
#include <QMdiArea>
#include <QMdiSubWindow>
#include <QPushButton>
#include <QHBoxLayout>

Widget::Widget(QWidget *parent)
: QWidget(parent)
{
    setWindowTitle(QString::fromUtf8("MDI Example"));
    setFixedSize(600,400);

    QMdiArea* area = new QMdiArea();
    area->setSizePolicy(QSizePolicy::Expanding,
                         QSizePolicy::Expanding);

    QMdiSubWindow* subWindow1 = new QMdiSubWindow();
    subWindow1->resize(300, 200);

    QPushButton *btn = new QPushButton(QString("Button"));
    subWindow1->setWidget(btn);

    QMdiSubWindow* subWindow2 = new QMdiSubWindow();
    subWindow2->resize(300, 200);

    area->addSubWindow(subWindow1);
    area->addSubWindow(subWindow2);

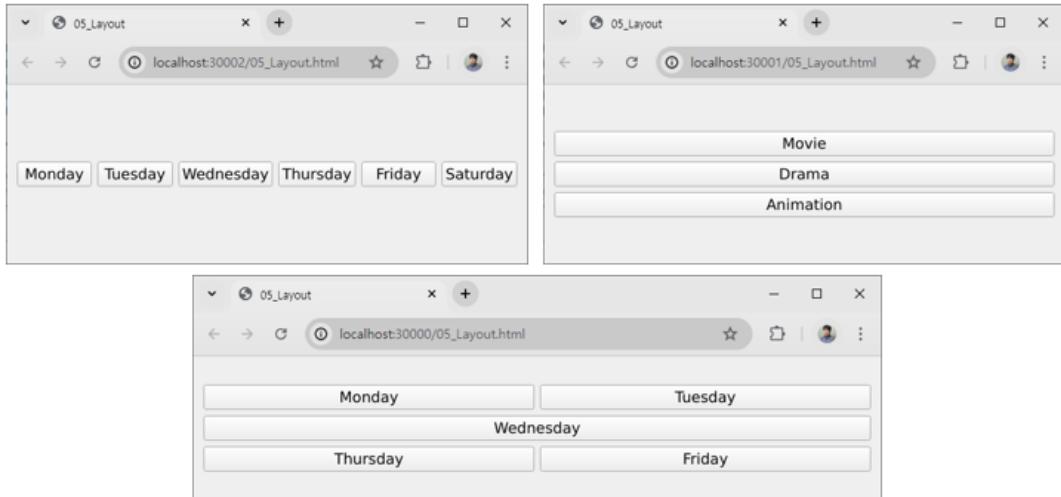
    QHBoxLayout *hboxLayout = new QHBoxLayout();
    hboxLayout->addWidget(area);

    setLayout(hboxLayout);
}
```

```
Widget::~Widget() {}
```

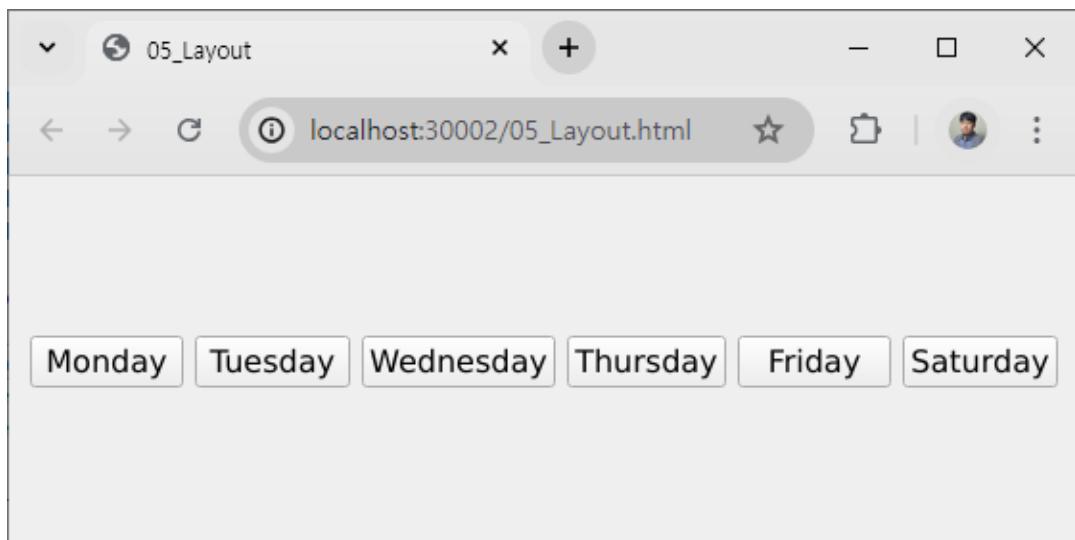
- 주로 사용하는 Layout

Qt는 다양한 Layout을 제공한다. 여기서는 주로 사용하는 Layout으로써 QHBoxLayout, QVBoxLayout 그리고 QGridLayout에 대해서 알아보도록 하자.



- QHBoxLayout

가로방향으로 Widget들을 배치한다.



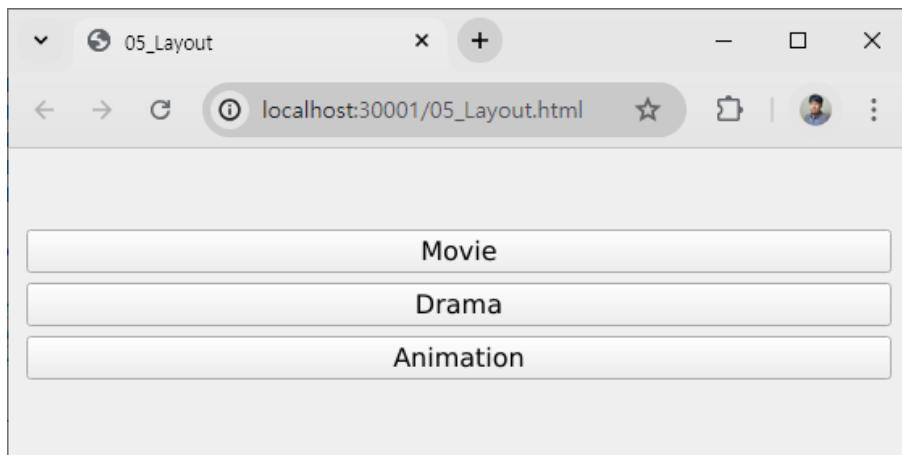
예수님은 당신을 사랑합니다.

```
...
QHBoxLayout *hboxLayout = new QHBoxLayout(this);
QPushButton *btn[6];
QString btnStr[6] = {"Monday", "Tuesday", "Wednesday", "Thursday",
                     "Friday", "Saturday"};

for( int i = 0 ; i < 6 ; i++ )
{
    btn[i] = new QPushButton(btnStr[i]);
    hboxLayout->addWidget(btn[i]);
}
setLayout(hboxLayout);
...
```

● QVBoxLayout

QVBoxLayout은 Widget들을 새로 방향으로 배치한다.



```
...
QVBoxLayout *vboxLayout = new QVBoxLayout();
QPushButton *vbtn[6];
```

예수님은 당신을 사랑합니다.

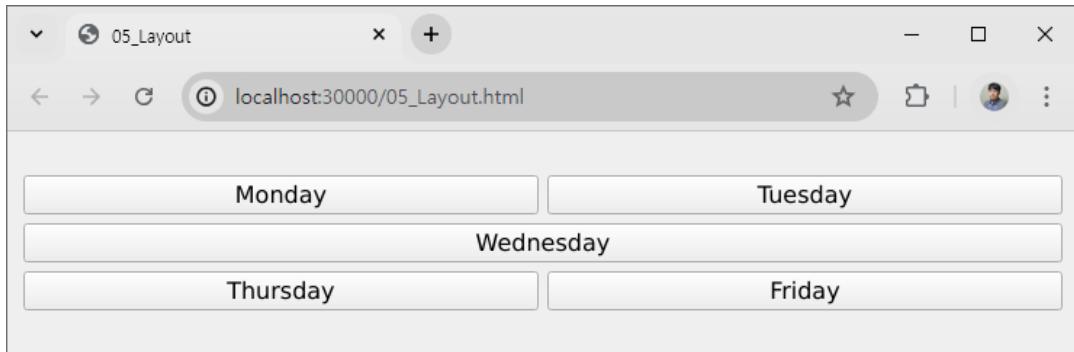
```
QString vbtnStr[3] = {"Movie", "Drama", "Animation"};
```

```
for(int i = 0 ; i < 3 ; i++)
{
    vbtn[i] = new QPushButton(vbtnStr[i]);
    vboxLayout->addWidget(vbtn[i]);
}
```

...

- QGridLayout

- 0) Layout은 Cell과 같이 Widget들을 배치한다.



...

```
QGridLayout *gridLayout = new QGridLayout();
QPushButton *gbtn[5];
QString gbtnStr[6] = {"Monday", "Tuesday", "Wednesday",
                      "Thursday", "Friday", "Saturday"};
for(int i = 0 ; i < 5 ; i++)
{
    gbtn[i] = new QPushButton(gbtnStr[i]);
```

```
gridLayout->addWidget(gbtn[0], 0, 0);
gridLayout->addWidget(gbtn[1], 0, 1);
gridLayout->addWidget(gbtn[2], 1, 0, 1, 2);
gridLayout->addWidget(gbtn[3], 2, 0);
gridLayout->addWidget(gbtn[4], 2, 1);

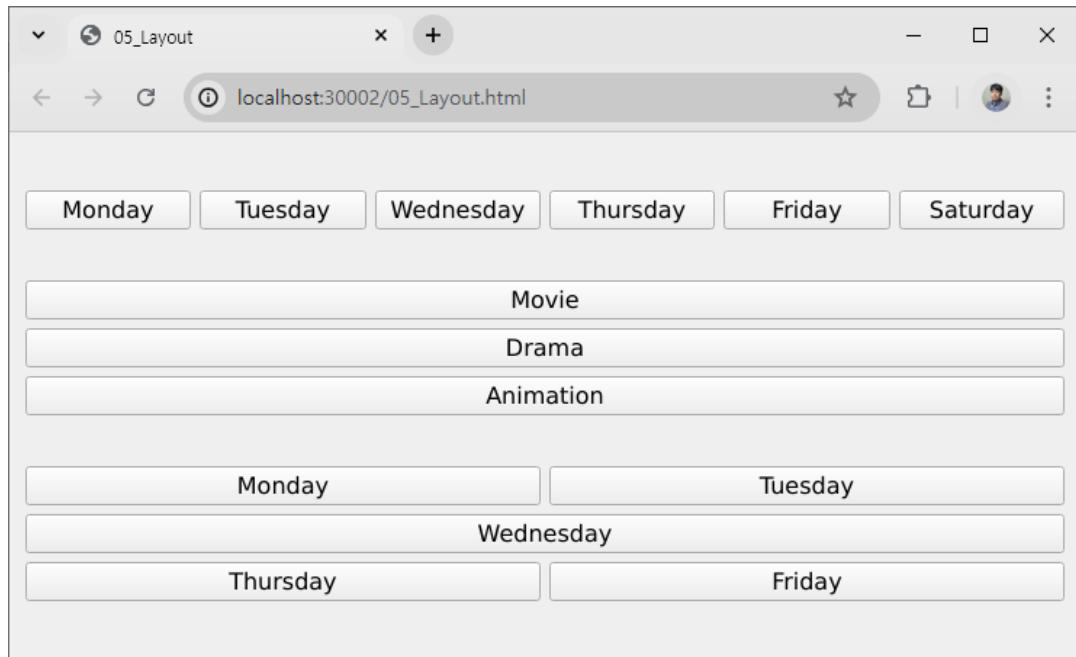
QVBoxLayout *defaultLayout = new QVBoxLayout();

defaultLayout->addLayout(gridLayout);
setLayout(defaultLayout);

...
```

- Layout들을 중첩된 구조로 사용하기

GUI를 구현할 때 다양한 Layout을 사용한다. 따라서 중첩된 구조의 Layout을 예로 들어 아래와 같은 GUI를 작성해 보도록 하자.



예수님은 당신을 사랑합니다.

```
#include "widget.h"
#include <QHBoxLayout>
#include <QVBoxLayout>
#include <QGridLayout>
#include <QPushButton>

Widget::Widget(QWidget *parent)
: QWidget(parent)
{
    QHBoxLayout *hboxLayout = new QHBoxLayout();
    QPushButton *btn[6];

    QString btnStr[6] = {"Monday", "Tuesday", "Wednesday",
                         "Thursday", "Friday", "Saturday"};

    for(int i = 0 ; i < 6 ; i++)
    {
        btn[i] = new QPushButton(btnStr[i]);
        hboxLayout->addWidget(btn[i]);
    }

    QVBoxLayout *vboxLayout = new QVBoxLayout();
    QPushButton *vbtn[6];

    QString vbtnStr[3] = {"Movie", "Drama", "Animation"};

    for(int i = 0 ; i < 3 ; i++)
    {
        vbtn[i] = new QPushButton(vbtnStr[i]);
        vboxLayout->addWidget(vbtn[i]);
    }
```

예수님은 당신을 사랑합니다.

```
QGridLayout *gridLayout = new QGridLayout();
QPushButton *gbtn[5];

QString gbtnStr[6] = {"Monday", "Tuesday", "Wednesday",
                      "Thursday", "Friday", "Saturday"};

for(int i = 0 ; i < 5 ; i++)
{
    gbtn[i] = new QPushButton(gbtnStr[i]);
}

gridLayout->addWidget(gbtn[0], 0, 0);
gridLayout->addWidget(gbtn[1], 0, 1);
gridLayout->addWidget(gbtn[2], 1, 0, 1, 2);
gridLayout->addWidget(gbtn[3], 2, 0);
gridLayout->addWidget(gbtn[4], 2, 1);

QVBoxLayout *defaultLayout = new QVBoxLayout();

defaultLayout->addLayout(hboxLayout);
defaultLayout->addLayout(vboxLayout);
defaultLayout->addLayout(gridLayout);

setLayout(defaultLayout);

}

Widget::~Widget() {}
```

8. 기본 데이터 타입과 유용한 타입들

소스코드를 구현할 때 변수의 타입을 선언할 때, int, float과 같은 타입을 사용한다. C/C++에서 사용하는 변수타입을 사용할 수 있다. 또한 Qt에서는 좀더 명확한 변수 타입의 이름을 아래의 표에서 보는 것과 같이 제공한다.

Type	Size (Bit)	Description
bool	8	true / false
qint8	8	signed char
qint16	16	signed short
qint32	32	signed int
qint64	64	long long int
quint8	8	unsigned char
quint16	16	unsigned short
quint32	32	unsigned int
quint64	64	unsigned long long int
float	32	floating point number
double	64	floating point number
const char*	32	문자열 상수를 가리키는 포인터, 마지막에 0은 제외

이외에도 더 많은 유용한 타입을 제공한다. "Qt Type Declarations" 키워드로 검색해보면 더 많은 정보를 얻을 수 있다.

Qt는 데이터 타입을 쉽게 핸들링 하기 위해서 다양한 Template 함수를 제공한다. 예를 들어 절대값을 구하기 위해서 qAbs() 함수를 제공하다. 최대값과 최소값 사이의 값을 구하기 위해서 qBound() 함수를 제공한다. 최대값과 최소 값을 구하기 위해서 qMax() 와 qMin() 함수를 제공한다.

- 절대 값 구하기

```
int absoluteValue;  
int myValue = -4;  
  
absoluteValue = qAbs(myValue); // absoluteValue == 4
```

- 최소값과 최대값 사이의 값을 구하기

```
int myValue = 10;  
int minValue = 2;  
int maxValue = 6;  
  
int boundedValue = qBound(minValue, myValue, maxValue);  
// boundedValue == 6
```

- 소수점을 비교하기 위한 함수

```
// 0.0과 비교  
qFuzzyCompare(0.0, 1.0e-200); // return false  
qFuzzyCompare(1 + 0.0, 1 + 1.0e-200); // return true
```

- 최대값 구하기

```
int myValue = 6;  
int yourValue = 4;  
int maxValue = qMax(myValue, yourValue);  
int minValue = qMin(myValue, yourValue);
```

- 반올림

```
qreal valueA = 2.3;  
qreal valueB = 2.7;
```

예수님은 당신을 사랑합니다.

```
int roundedValueA = qRound(valueA); // roundedValueA = 2  
int roundedValueB = qRound(valueB); // roundedValueB = 3  
  
valueA = 42949672960.3;  
valueB = 42949672960.7;  
roundedA = qRound(valueA); // roundedA = 42949672960  
roundedB = qRound(valueB); // roundedB = 42949672961
```

Qt는 단순한 문자열을 다루는 클래스 이외에도 데이터 스트림, 멀티 바이트 캐릭터 형태의 유니코드 4.0(Unicode Standard Version) 캐릭터를 지원하는 다양한 클래스를 제공한다.

Class name	Description
QBitArray	Bit 단위 데이터 조작 기능과 AND, OR, NOT 에 대한 연산자 제공.
QByteArray	1 Byte 단위의 배열 형태
QString	문자열을 저장, 한 문자열을 저장하는데 QChar (16-bit Unicode character) 를 사용.
QVariant	Void 타입과 같이 정해져 있지 않은 타입을 다룰 때 유용하게 사용이 가능.
QPoint	X, Y 값을 저장하는데 사용 QPointF는 소수점을 저장할 때 사용.
QRect	X, Y, WIDTH, HEIGHT 값을 저장하는 데 사용. QRectF는 소수점을 저장할 때 사용.
QRegExp	정규화 표현식을 처리하기 위한 클래스
QRegion	특정 영역이 겹치는지 체크하기 위한 용도로 사용, 예를 들어 집합에서 합집합 영역, 교집합 영역 등.
QSize	넓이와 높이를 저장할 수 있는 클래스
QVariant	void 타입으로 저장할 수 있는 union 형태의 클래스
QVector2D	2차원 공간에서 Vector 또는 vertex를 나타내기 위한 클래스

QVector3D	X, Y, Z 좌표를 사용할 수 있는 클래스
QVector4D	4차원 공간에서 Vector 또는 vertex를 나타내기 위해서 사용

- QBitArray

QBitArray는 Bit 단위의 데이터를 핸들링 하기 위한 목적으로 사용한다. 200개의 Bit를 선언하기 위해서 아래와 같이 사용할 수 있다.

```
QBitArray ba(200);
```

동적으로 크기를 선언하기 위해서 아래와 같이 resize() 멤버함수를 사용할 수 있다.

```
QBitArray ba;  
ba.resize(3);  
ba[0] = true;  
ba[1] = false;  
ba[2] = true;
```

특정 Bit의 값을 변경하기 위해서 setBit() 함수를 사용하면 된다.

```
QBitArray ba(3);  
ba.setBit(0, true);  
ba.setBit(1, false);  
ba.setBit(2, true);
```

연사자를 사용하고자 할 때 아래와 같이 사용할 수 있다.

- QByteArray

QByteArray 클래스는 바이트(8-bit) 단위의 배열을 제공한다. QByteArray 클래스는 배열을 핸들링 하기 위해 append(), prepend(), insert(), replace() 그리고 remove() 멤버 함수를 제공한다.

```
QByteArray x("Q");  
  
x.prepend("I love"); // x == I love Q  
x.append("t -^ ^*"); // x == I love Qt -^ ^*
```

예수님은 당신을 사랑합니다.

```
x.replace(13, 1, "*"); // x == I love Qt *^_^*
```

```
QByteArray x("I love Qt -^_^*");
x.remove(13, 4); // x == I love Qt
```

- QByteArray

1Byte 단위(8bits)로 저장된다. 문자열, Hex 등 다양하게 저장할 수 있다. 그리고 크기를 동적으로 추가, 수정 그리고 삭제를 편리하게 할 수 있는 멤버 함수를 제공한다.

```
QByteArray ba("Hello");
```

동적으로 사이즈를 지정하기 위해서 resize() 함수를 사용하면 된다.

```
QByteArray ba;
ba.resize(5);
ba[0] = 0x3c;
ba[1] = 0xb8;
ba[2] = 0x64;
ba[3] = 0x18;
ba[4] = 0xca;
```

특정 주소의 배열을 접근하기 위해서 at() 멤버 함수를 사용한다.

```
QByteArray ba("aAHeLLo, ");
for (int i = 0; i < ba.size(); ++i)
{
    if (ba.at(i) >= 'a' && ba.at(i) <= 'f')
        qDebug() << "Found character in range [a-f] : " << ba.at(i) ;
}
```

현재 저장된 문자열 앞에 붙여넣기 하기 위해서 prepend() 함수를 사용하면 된다. 그리고 문자열 뒤에 붙이기 위해서는 append() 함수를 사용하면 된다. 문자열에서 특정

예수님은 당신을 사랑합니다.

문자열을 변경하기 위해서 replace() 함수를 사용하면 된다.

```
QByteArray x("and");

x.prepend("rock ");           // x == "rock and"
x.append(" roll");          // x == "rock and roll"
x.replace(5, 3, "&");       // x == "rock & roll"
```

QByteArray에 저장된 문자열 중 특정 문자열을 찾기 위해서 다음과 같이 사용할 수 있다.

```
QByteArray ba("abc <b>bold</b>, def <b>bold</b>");
qsize type j = 0;
while ((j = ba.indexOf("<b>", j)) != -1)
{
    qDebug() << "<b> Position : " << j;
    ++j;
}
```

char * 으로 변환하기 위해서 data() 함수를 아래와 같이 사용할 수 있다.

```
QByteArray ba("Hello");

char *data = ba.data();

while (*data) {
    qDebug() << "[" << *data << "]";
    ++data;
}

// [ H ]
// [ e ]
// [ l ]
// [ l ]
```

예수님은 당신을 사랑합니다.

```
//[ o ]
```

fill() 멤버 함수를 사용하기 위해서 아래와 같이 사용할 수 있다.

```
QByteArray ba("Istambul");
```

```
ba.fill('o');
```

```
ba.fill('X', 2);
```

```
// [Result] ba == "oooooooo", // ba == "XX"
```

first() 함수를 사용하면 아래와 같이 지정한 수의 문자를 추출할 수 있다.

```
QByteArray x("Pineapple");
```

```
QByteArray y = x.first(4);
```

```
// y == "Pine
```

Base64 포맷의 데이터를 Decode 하기 위해서 아래와 같이 사용할 수 있다.

```
QByteArray text = QByteArray::fromBase64("UXQgaXMgZ3JlYXQh");
```

```
qDebug() << "문자열 :" << text.data();
```

Hex로 저장된 데이터를 변환 할 수 있다. URL/URI 스타일의 입력된 Encoding 된 데이터를 변환할 수 있는 기능을 제공한다.

```
QByteArray text = QByteArray::fromHex("517420697320677265617421");
```

```
qDebug() << "String :" << text.data();
```

```
QByteArray text = QByteArray::fromPercentEncoding("Qt%20is%20great%33");
```

```
qDebug() << "String :" << text.data();
```

```
// String : Qt is great!
```

```
// String : Qt is great!
```

문자열을 int, float, long 등과 같은 타입으로 변환하여 저장할 수 있는 기능을 제공한다.

```
QByteArray str("FF");
```

```
bool ok;
```

```
int hex = str.toInt(&ok, 16);
int dec = str.toInt(&ok, 10);

// hex == 255, ok == true
// dec == 0, ok == false
```

- **QByteArrayMatcher**

바이트 배열에서 매칭되는 바이트 배열 패턴을 찾기 위해 제공되는 클래스이다.

```
// 전체 QByteArray
QByteArray x(" hello Qt, nice to meet you.");
QByteArray y("Qt"); // x에서 찾고자 하는 문자열

QByteArrayMatcher matcher(y);

//문자열이 시작되는 위치 index 변수에 저장
int index = matcher.indexIn(x, 0);

qDebug( "index : %d", index);
qDebug( "QByte : %c%c", x.at(index), x.at(index+1));
```

- **Qchar**

16Bit 유니코드를 지원하기 위한 Character클래스이다.

```
QLabel *lbl = new QLabel("", this);
QString str = "Hello Qt";
QChar *data = str.data();
QString str_display;

while(!data->isNull())
```

예수님은 당신을 사랑합니다.

```
{  
    str_display.append(data->unicode());  
    ++data;  
}  
  
lbl->setText(str_display); // Hello Qt
```

- **QLatin1String**

US-ASCII/Latin-1 인코딩 문자열을 지원하기 위해서 제공되는 클래스이다.

```
QLatin1String latin("Qt");  
QString str = QString("Qt");  
  
if(str == latin)  
    qDebug("Equal.");  
else  
    qDebug("Not equal.");  
  
bool is_equal = latin.operator==(str);  
  
if(is_equal)  
    qDebug("Equal.");  
else  
    qDebug("Not equal.");
```

- **QLocale**

이 클래스는 다양한 언어 Character set으로 변환하기 위해 사용한다.

```
QLocale egyptian(QLocale::Arabic, QLocale::Egypt);  
QString s1 = egyptian.toString(1.571429E+07, 'e');  
QString s2 = egyptian.toString(10);
```

예수님은 당신을 사랑합니다.

```
double d = egyptian.toDouble(s1);
int i = egyptian.toInt(s2);
```

- `QString`

`QString` 클래스는 유니코드 문자열을 지원하며 16bit `QChar`를 저장할 수 있는 기능 제공한다.

```
QString str = "Hello";
```

`QString` 은 `const char *` 와 같은 문자열 상수를 `fromUtf8()` 함수를 사용해 대체할 수 있는 기능을 제공한다.

```
static const QChar data[4] = {0x0055, 0x006e, 0x10e3, 0x03a3 };
QString str(data, 4);
```

저장된 문자열의 특정 위치에 `QChar`을 저장할 수 있는 기능 제공한다.

```
QString str;
str.resize(4);

str[0] = QChar('U');
str[1] = QChar('n');
str[2] = QChar(0x10e3);
str[3] = QChar(0x03a3);
```

문자열을 비교하기 위해 `QString` 은 `if` 문을 이용해 다음과 같은 비교가 가능하다.

```
QString str;
if ( str == "auto" || str == "extern" || str == "static" || str == "register" )
{
    // ...
}
```

찾고자 하는 특정 문자열의 위치를 알고 싶다면 `indexOf()` 함수를 사용해 찾고자 하는 문자열의 위치를 찾을 수 있다.

```
QString str = "We must be <b>bold</b>, very <b>bold</b>";
```

예수님은 당신을 사랑합니다.

```
int j = 0;

while ((j = str.indexOf("<b>", j)) != -1) {
    qDebug() << "Found <b> tag at index position" << j;
    ++j;
}
```

QString은 arg() 함수를 이용해 특정 타입을 함께 사용할 수 있다.

```
int value;
QByteArray word;
QString sentence;

value = 14;
word = QByteArray("Qt");
sentence = QString("Hello World");

QString status = QString("%1, %2, %3").arg(value).arg(word).arg(sentence);
qDebug() << status;
```

chop() 함수를 통해서 아래와 같은 \r\n을 제거할 수 있다.

```
QString str("LOGOUT\r\n");
str.chop(2);
```

- QString의 비교와 Regular Expression 사용

```
int a = QString::compare("aUtO", "AuTo", Qt::CaseInsensitive); // a == 0
int b = QString::compare("aUtO", "AuTo", Qt::CaseSensitive); // b > 0
int c = QString::compare("AuTo", "AuTo", Qt::CaseSensitive); // c == 0
```

예를 들어 "Peter Pan" 문자열에 "peter"가 포함되어 있는지 검사하기 위해서 아래와 같이 사용할 수 있다.

```
QString str = "Peter Pan";
bool ret = str.contains("peter", Qt::CaseInsensitive);
```

예수님은 당신을 사랑합니다.

만약 대소문자도 구별해야 한다면 다음과 같이 사용할 수 있다.

```
QString str = "Peter Pan";
bool ret = str.contains("peter", Qt::CaseSensitive);
```

Regular Expression을 사용하기 위해서 아래와 같이 사용할 수 있다.

```
QString str = "banana and panama";
str.count(QRegularExpression("a[nm]a"));
```

문자열의 끝에 특정 문자열이 포함되어 있는지 검사하기 위해서 다음과 같이 사용할 수 있다.

```
QString str = "Bananas";
str.endsWith("anas");
str.endsWith("pple");

// anas: true, pple: false
```

QString에서 공백을 제거하기 위해서 trimmed() 함수를 사용할 수 있다.

```
QString str = " lots of whitespace ";
str = str.trimmed();
```

- QVariant

타입이 정해지지 않은 타입을 사용할 수 있다.

```
int original = 123;
QVariant v(original);

int x = v.toInt();
qDebug() << "x : " << x;

QString xStr = v.toString();
qDebug() << "xStr : " << xStr;
qDebug() << "v 의 타입명 :" << v.typeName();
```

예수님은 당신을 사랑합니다.

- QPoint

Integer 값의 X, Y 값을 저장하는데 사용한다.

```
QPoint p( 3, 7);
QPoint q(-1, 4);

p += q;    // p becomes (2, 11)
```

QPoint는 rx() 함수와 ry() 함수를 아래와 같이 사용할 수 있다.

```
QPoint p(1, 2);
p.rx()--; // p becomes (0, 2)

QPoint p(1, 2);
p.ry()++; // p becomes (1, 3)
```

아래와 같은 연산자를 사용할 수 있다.

```
QPoint p( 3, 7);
QPoint q(-1, 4);

p -= q;    // p becomes (4, 3)
```

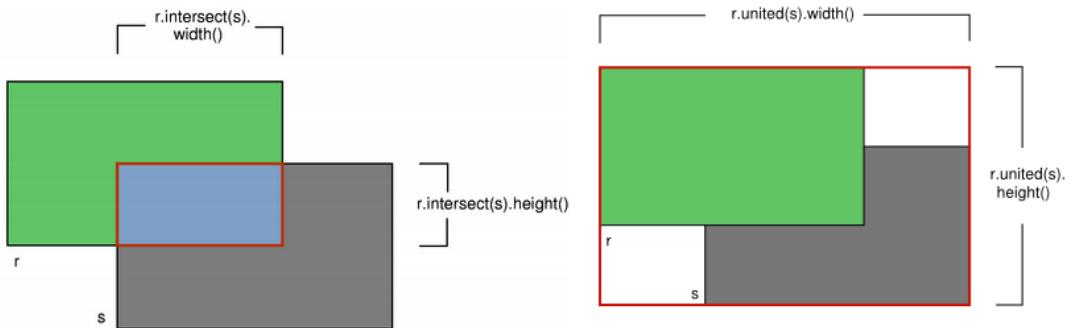
- QRect

QRect는 X, Y, WIDTH, HEIGHT 값을 저장하는데 유용하게 사용할 수 있다.

```
QRect r1(100, 200, 11, 16);
QRect r2(QPoint(100, 200), QSize(11, 16));
```

그리고 QRect는 intersect() 함수와 united() 함수를 이용해 교집합영역과 합집합영역을 구할 수 있다.

예수님은 당신을 사랑합니다.



9. Qt에서 제공하는 유용한 Template class

Qt에서 제공하는 Template 클래스들은 STL에서 제공하는 Container들보다 사용하기 쉽고 안전하다. 또한 경량화 되어 있다. 따라서 Qt에서 제공하는 Template class들은 C++의 Template 보다 쉽게 사용할 수 있다.

- QMap

QMap은 Key 기반의 빠른 검색을 제공하며 Key와 Value값이 pair로 저장된다.

```
QMap<QString, int> map;
```

Key와 Value를 QMap에 추가하는 방법은 아래와 같이 사용할 수 있다.

```
map["one"] = 1;  
map["three"] = 3;  
map["seven"] = 7;
```

위의 방법 이외에도 insert() 함수를 이용하면 Key와 Value를 추가할 수 있다.

```
map.insert("twelve", 12);
```

특정 Key값에 맵핑 된 값을 참조하기 위해서 다음과 같이 사용할 수 있다.

```
int num1 = map["thirteen"];  
int num2 = map.value("thirteen");
```

특정 Key 값이 존재하는지 contains()함수를 사용할 수 있다.

```
int timeout;  
if (map.contains("TIMEOUT"))  
    timeout = map.value("TIMEOUT");
```

- QHash

QHash 클래스는 해시 테이블 기반의 Dictionary를 제공한다. 데이터를 저장하는 방식은 Key, Value 가 Pair(쌍)로 저장된다. Key 값으로 찾고자 하는 데이터를 빠르게 검색

예수님은 당신을 사랑합니다.

할 수 있는 기능을 제공한다. QHash는 QMap과 매우 비슷한 기능을 제공하지만 내부 알고리즘은 QMap 보다 빠르다.

```
QHash<QString, int> hash;
```

```
hash["one"] = 1;  
hash["three"] = 3;  
hash["seven"] = 7;
```

QHash에 Key, Value를 쌍으로 저장하기 위한 방법으로 insert() 함수를 사용할 수 있다. 그리고 Value 값을 알기 위해 value() 멤버 함수를 사용할 수 있다.

```
hash.insert("twelve", 12);  
int num1 = hash["thirteen"];  
int num2 = hash.value("thirteen");
```

● QPair

QPair 클래스는 두 개의 아이템을 하나로 구성된 Pair로 저장할 수 있다. 아래 예제에서 보는 것과 같이 QPair 클래스는 다음과 같이 선언하여 사용할 수 있다

```
QPair<QString, double> pair;  
  
pair.first = "pi";  
pair.second = 3.14159265358979323846;
```

● QList

QList<T>는 빠른 인덱스 기반의 액세스가 가능하며 저장된 데이터 삭제도 매우 빠르다. QList는 인덱스 기반의 클래스이며 QLinkedList의 Iterator 기반보다 사용하기 편리하다.

```
QList<int> integerList;  
QList<QDate> dateList;  
QList<QString> list = { "one", "two", "three" }
```

QList는 비교 연산자를 통해 리턴 값을 아래 예와 같이 사용할 수 있다.

예수님은 당신을 사랑합니다.

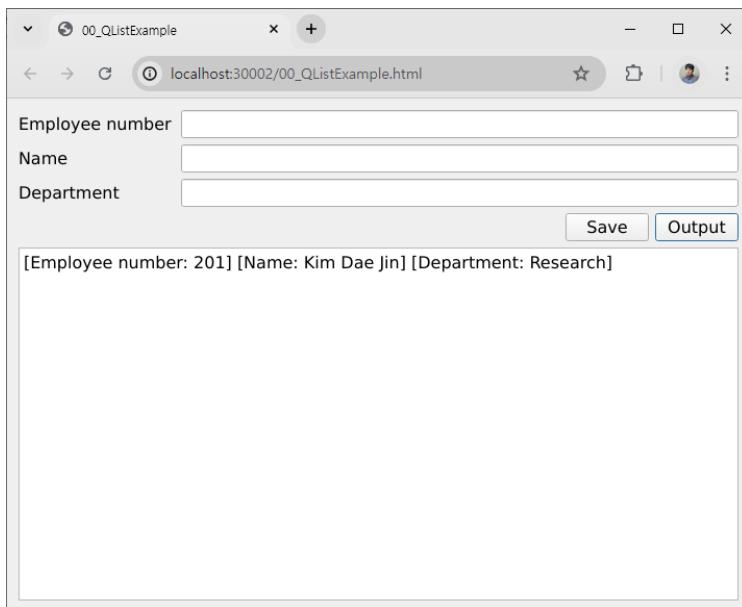
```
if (list[0] == "Bob")
    list[0] = "Robert";
```

QList는 at() 함수를 이용해 리스트 상에 저장된 위치를 쉽게 검색 할 수 있다.

```
for (int i = 0 ; i < list.size() ; ++i) {
    if (list.at(i) == "Jane")
        cout << "Found Jane at position " << i << endl;
}
```

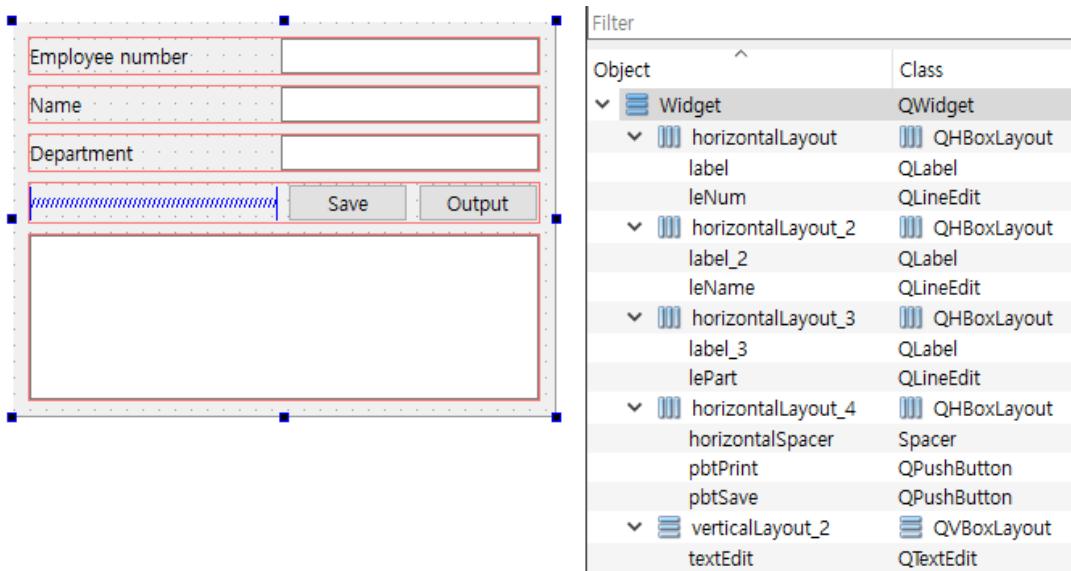
● QList와 Structure를 이용한 예제

이번에는 QList와 Structure를 이용해 아래와 같은 예제를 작성해 보도록 하자.



Widget기반의 프로젝트를 생성한 그런 다음 Widget.ui 파일을 더블 클릭해 아래와 같이 GUI Widget들을 배치한다.

예수님은 당신을 사랑합니다.



다음으로 widget.h 헤더파일을 열어서 아래와 같이 소스코드를 작성한다.

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>

QT_BEGIN_NAMESPACE
namespace Ui {
class Widget;
}
QT_END_NAMESPACE

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();
}
```

예수님은 당신을 사랑합니다.

```
private:  
Ui::Widget *ui;  
  
typedef struct _tEmployee {  
    int num;  
    QString name;  
    QString part;  
} tEmployee;  
  
QList<tEmployee> m_employeeList;  
  
private slots:  
void slot_pbtSave();  
void slot_pbtPrint();  
  
};  
#endif // WIDGET_H
```

다음으로 widget.cpp 소스코드 파일을 아래와 같이 작성한다.

```
#include "widget.h"  
#include "./ui_widget.h"  
#include <QFontDatabase>  
  
Widget::Widget(QWidget *parent)  
: QWidget(parent), ui(new Ui::Widget)  
{  
    ui->setupUi(this);  
  
    connect(ui->pbtSave, SIGNAL(pressed()), this, SLOT(slot_pbtSave()));  
    connect(ui->pbtPrint, SIGNAL(pressed()), this, SLOT(slot_pbtPrint()));  
}  
  
void Widget::slot_pbtSave()
```

예수님은 당신을 사랑합니다.

```
{  
    int num = ui->leNum->text().toInt();  
    QString name = ui->leName->text();  
    QString part = ui->lePart->text();  
  
    tEmployee employee;  
    employee.num = num;  
    employee.name = name;  
    employee.part = part;  
  
    m_employeeList.append(employee);  
  
    ui->leNum->clear();  
    ui->leName->clear();  
    ui->lePart->clear();  
}  
  
void Widget::slot_pbtPrint()  
{  
    ui->textEdit->clear();  
  
    for( qsizetype i = 0 ; i < m_employeeList.size() ; i++ )  
    {  
        int num = m_employeeList.at(i).num;  
        QString name = m_employeeList.at(i).name;  
        QString part = m_employeeList.at(i).part;  
  
        QString str;  
        str = QString("[Employee number: %1] [Name: %2] [Department: %3]")  
            .arg(num).arg(name, part);  
  
        ui->textEdit->append(str);  
}
```

예수님은 당신을 사랑합니다.

```
    }  
}  
  
Widget::~Widget()  
{  
    delete ui;  
}
```

10. QPainter 클래스를 이용한 2D Graphics

이번장에서는 QPainter클래스를 이용해 QWidget상에 2D 그래픽 요소를 랜더링 하는 방법에 대해서 알아보도록 하자.

QWidget상에 2D 그래픽 요소를 랜더링 하기 위해서는 paintEvent() virtual 함수를 사용 해야 한다.

```
#ifndef WIDGET_H
#define WIDGET_H
#include <QWidget>
class Widget : public QWidget
{
    Q_OBJECT
public:
    Widget(QWidget *parent = nullptr);
    ~Widget();
protected:
    virtual void paintEvent(QPaintEvent *event);
};

#endif // WIDGET_H
```

위의 예제 헤더파일에서 보는 것과 같이 paintEvent() 함수를 선언한다. 그리고 소스코드 파일에서 아래와 같이 사용할 수 있다.

```
#include "widget.h"
#include <QPainter>

Widget::Widget(QWidget *parent) : QWidget(parent)
{}
```

```
void Widget::paintEvent(QPaintEvent *event)
{
    Q_UNUSED(event)
    QPainter painter;
    painter.begin(this);
    painter.setPen(Qt::blue);
    painter.drawLine(10, 10, 100, 40);
    painter.drawRect(120, 10, 80, 80);
    painter.end();
}
```

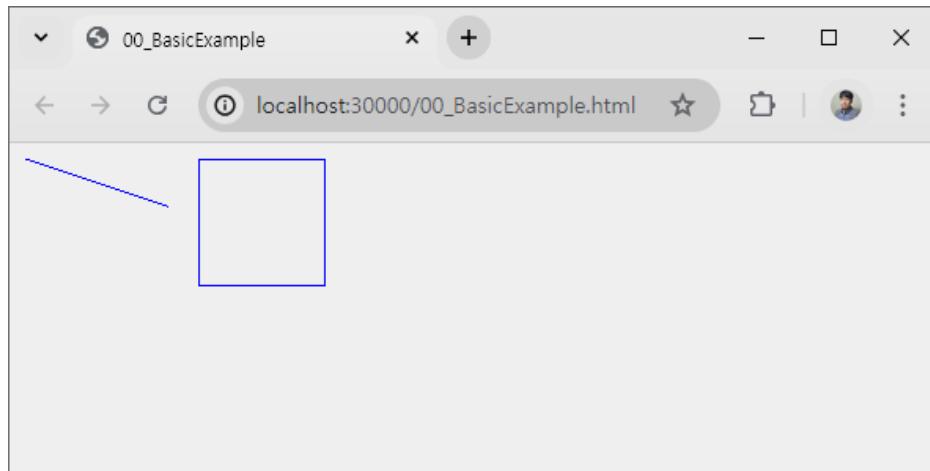
...

QPainter 클래스의 begin() 함수를 드로잉을 시작한다는 것을 뜻한다. 그리고 end()는 드로잉을 마친다는 뜻이다.

그리고 drawLine() 함수는 Line을 드로잉 한다. 첫 번째와 두 번째 인자는 X, Y좌표이다. 세 번째와 네 번째는 X2, Y2 좌표이다.

drawRect() 함수는 사각형을 드로잉 한다. 첫 번째, 두 번째 인자는 X, Y이고 세 번째는 WIDTH이다. 마지막 인자는 HEIGHT이다.

위의 소스코드를 실행하면 아래와 같이 Web Browser상에 실행되는 것을 확인할 수 있다.



- Line의 Style

QPainter영역에서 Line의 Style을 정의하기 위해서는 QPainter클래스의 setPen()멤버함수를 사용하면 된다.

```
...
void Widget::paintEvent(QPaintEvent *event)
{
    QPainter painter;
    painter.begin(this);

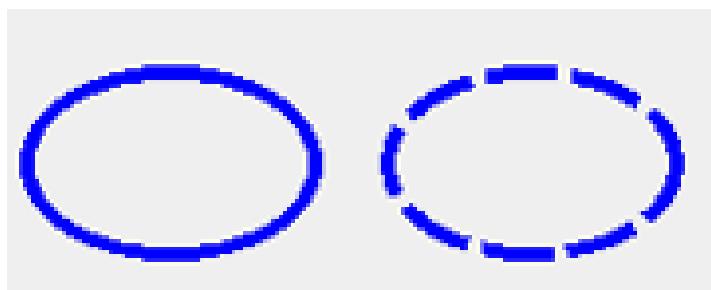
    QPen pen(Qt::blue);

    pen.setWidth(4);
    painter.setPen(pen);
    QRect rect1(10.0, 20.0, 80.0, 50);
    painter.drawEllipse(rect1);

    pen.setStyle(Qt::DashLine);
    painter.setPen(pen);
    QRect rect2(110.0, 20.0, 80.0, 50.0);

    painter.drawEllipse(rect2);
    painter.end();
}
```

...



- Line의 Join style

Line이 연결될 때 모서리의 Style을 다르게 사용하기 위해서 QPen의 setJoinStyle() 멤버함수를 사용하면 된다.

```
void Widget::paintEvent(QPaintEvent *event)
{
    QPainter painter;
    painter.begin(this);
    QPen pen(Qt::blue);
    pen.setWidth(10);
    QPointF p1[3] = {QPointF(30.0, 80.0),QPointF(20.0, 40.0), QPointF(80.0, 60.0 )};
    pen.setJoinStyle(Qt::BevelJoin);
    painter.setPen(pen);
    painter.drawPolyline(p1, 3);
    QPointF p2[3] = {QPointF(130.0, 80.0), QPointF(120.0, 40.0), QPointF(180.0, 60.0 )};
    pen.setJoinStyle(Qt::MiterJoin);
    painter.setPen(pen);
    painter.drawPolyline(p2, 3);
    QPointF p3[3] = {QPointF(230.0, 80.0), QPointF(220.0, 40.0), QPointF(280.0, 60.0 )};
    pen.setJoinStyle(Qt::RoundJoin);
    painter.setPen(pen);
    painter.drawPolyline(p3, 3);
    painter.end();
}
```



- QPainter의 setBrush()

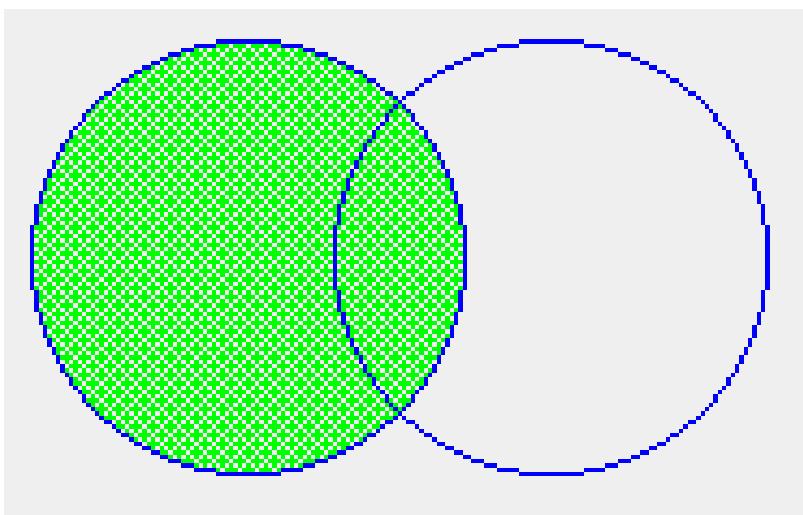
QPainter의 setBrush()함수를 이용해 원이나 사각형 모양 안에 특정 Brush를 이용할 수 있다.

```
void Widget::paintEvent(QPaintEvent *event)
{
    QPainter painter;
    painter.begin(this);

    painter.setBrush(QBrush(Qt::green, Qt::Dense3Pattern));
    painter.setPen(Qt::blue);
    painter.drawEllipse(10, 10, 100, 100);

    painter.setBrush(Qt::NoBrush);
    painter.setPen(Qt::blue);
    painter.drawEllipse(80, 10, 100, 100);

    painter.end();
}
```



- QPixmap

QPixmap클래스를 이용해 이미지를 QWidget상에 표시할 수 있다. 이미지를 표시하기 위해서 QImage등과 같은 클래스를 이용할 수 있다.

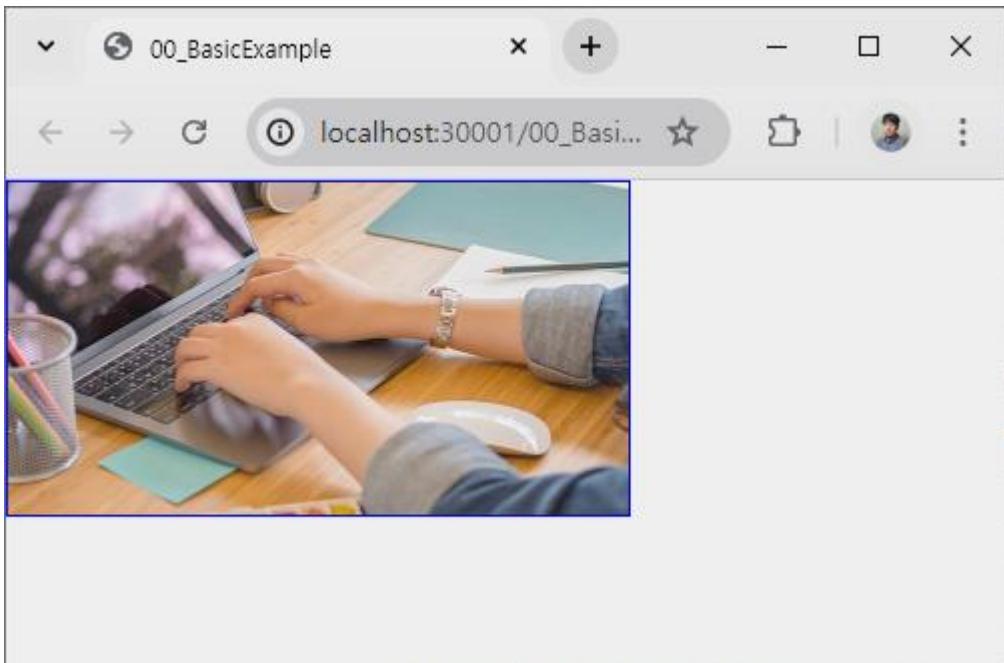
```
void Widget::paintEvent(QPaintEvent *event)
{
    QPainter painter;
    painter.begin(this);

    QPixmap pixmap(":/res/image.png");

    int w = pixmap.width();
    int h = pixmap.height();
    pixmap.scaled(w, h, Qt::IgnoreAspectRatio, Qt::SmoothTransformation);

    QBrush brush(pixmap);
    painter.setBrush(brush);
    painter.setPen(Qt::blue);
    painter.drawRect(0, 0, w, h);

    painter.end();
}
```



- **QTransform**

이 클래스를 이용하면 이동, 확대/축소, 회전, 원근표현 기법 등을 사용할 수 있다.

```
void Widget::paintEvent(QPaintEvent *event)
{
    QPainter painter;
    painter.begin(this);

    QImage image(":/images/image.png");

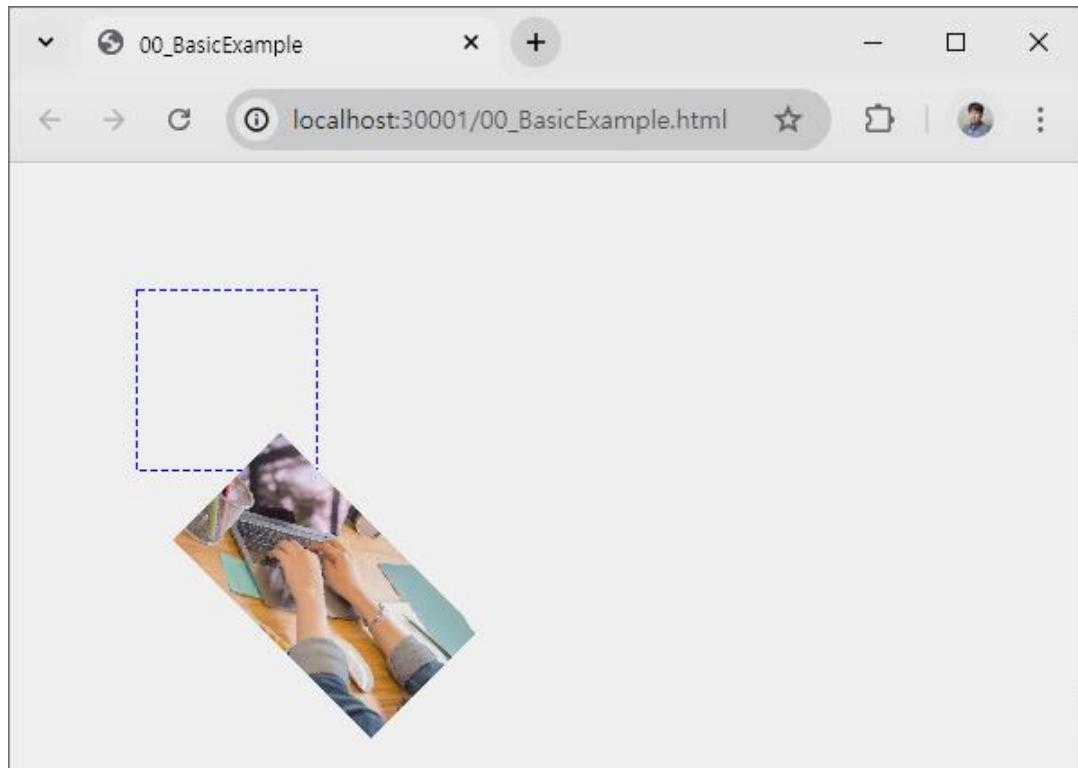
    painter.setPen(QPen(Qt::blue, 1, Qt::DashLine));
    painter.drawRect(70, 70, 100, 100);

    QTransform transform;
    transform.translate(150, 150);
    transform.rotate(45);
    transform.scale(0.5, 0.5);
```

예수님은 당신을 사랑합니다.

```
painter.setTransform(transform);
painter.drawImage(0, 0, image);

painter.end();
}
```



QTransform 클래스의 rotate() 함수를 이용하면 X축, Z축 또는 Z축의 원근 표현을 이용할 수 있다.

```
void Widget::paintEvent(QPaintEvent *event)
{
    QPainter painter;
    painter.begin(this);

    QImage image(":/images/image.png");
```

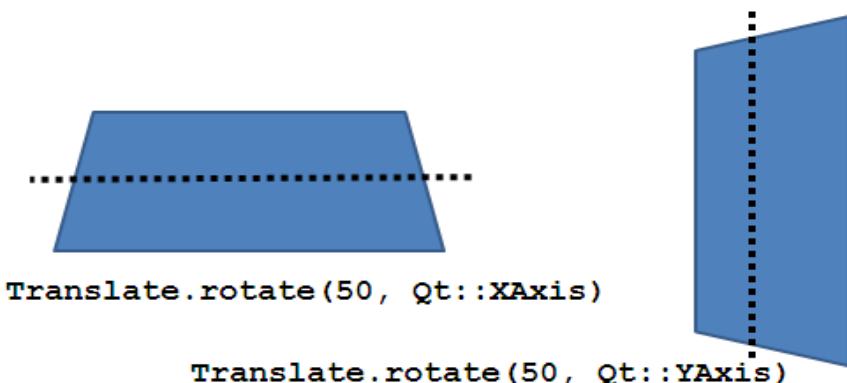
예수님은 당신을 사랑합니다.

```
painter.setPen(QPen(Qt::blue, 1, Qt::DashLine));
painter.drawRect(0, 0, 100, 100);

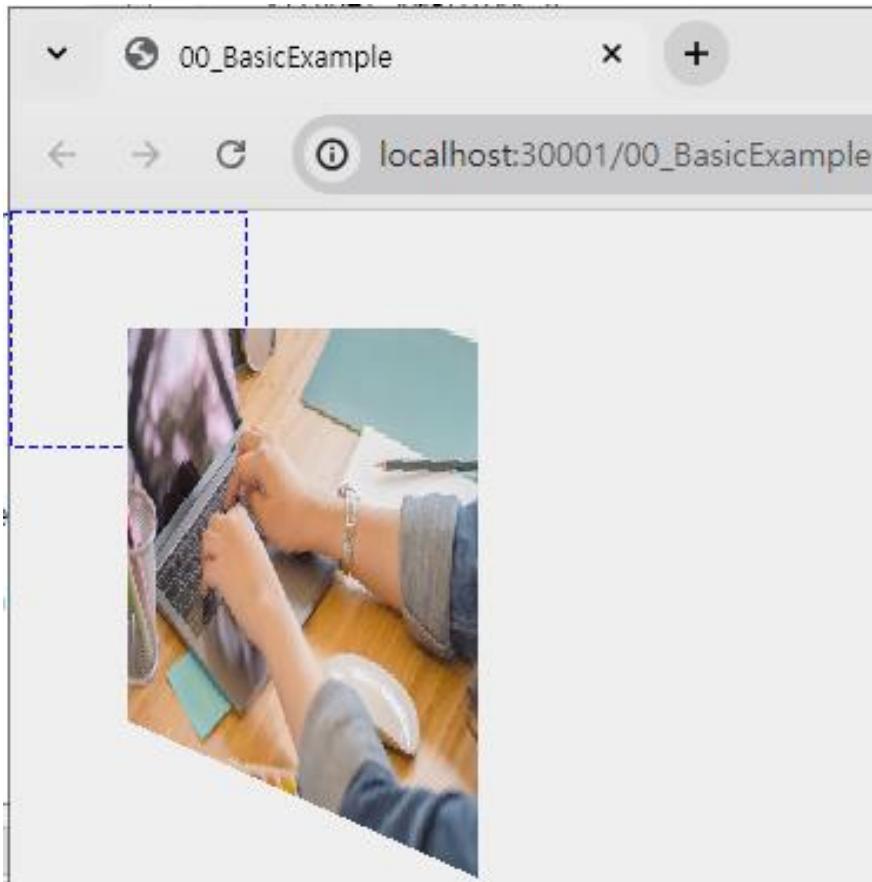
QTransform transform;
transform.translate(50, 50);
transform.rotate(70, Qt::YAxis);

painter.setTransform(transform);
painter.drawImage(0, 0, image);

painter.end();
}
```

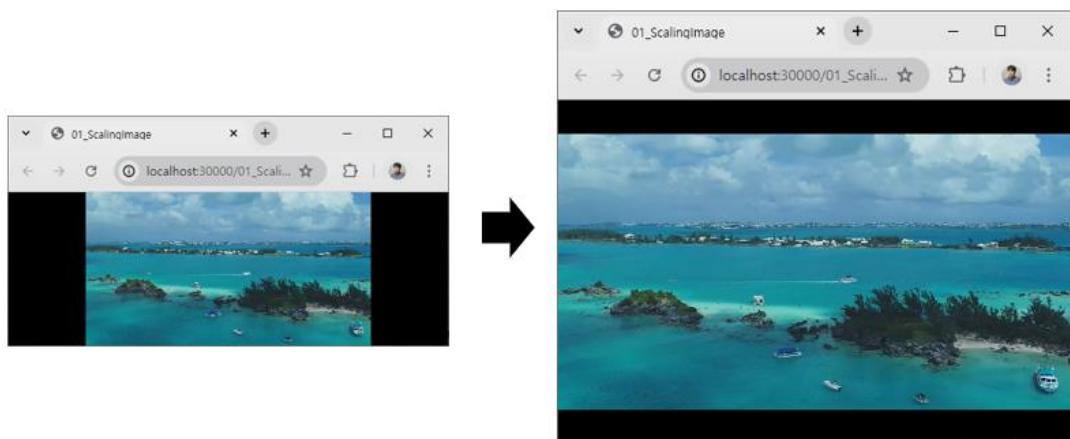


예수님은 당신을 사랑합니다.



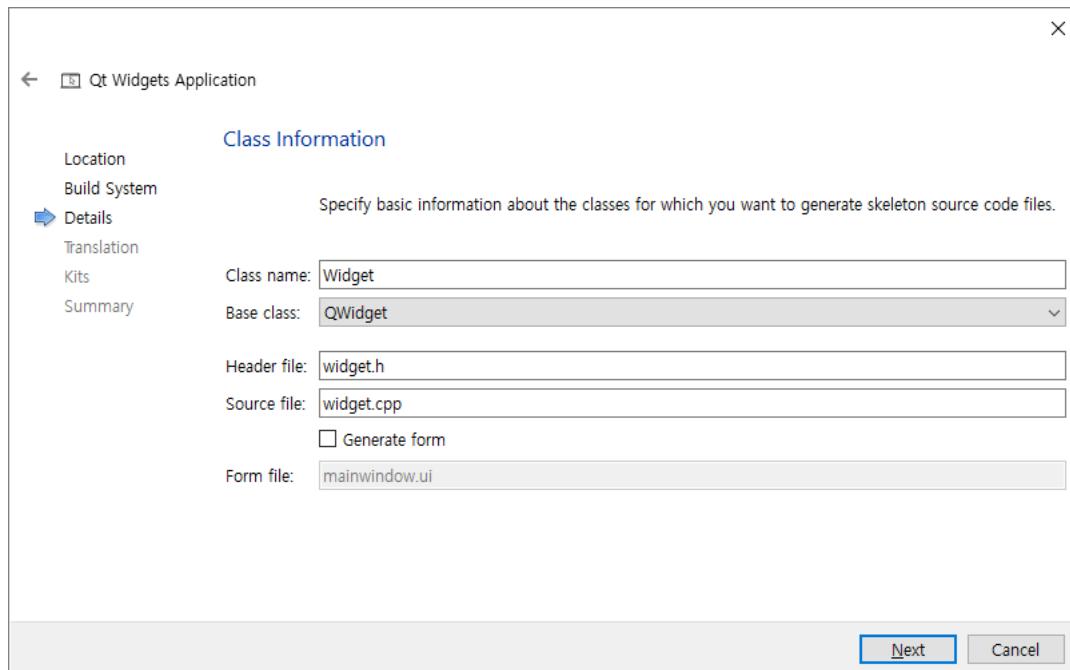
- Image Scaling

이번에는 QPainter클래스를 이용해 이미지크기를 Web Browser 크기에 따라 Scaling하는 예제를 작성해 보도록 하자.

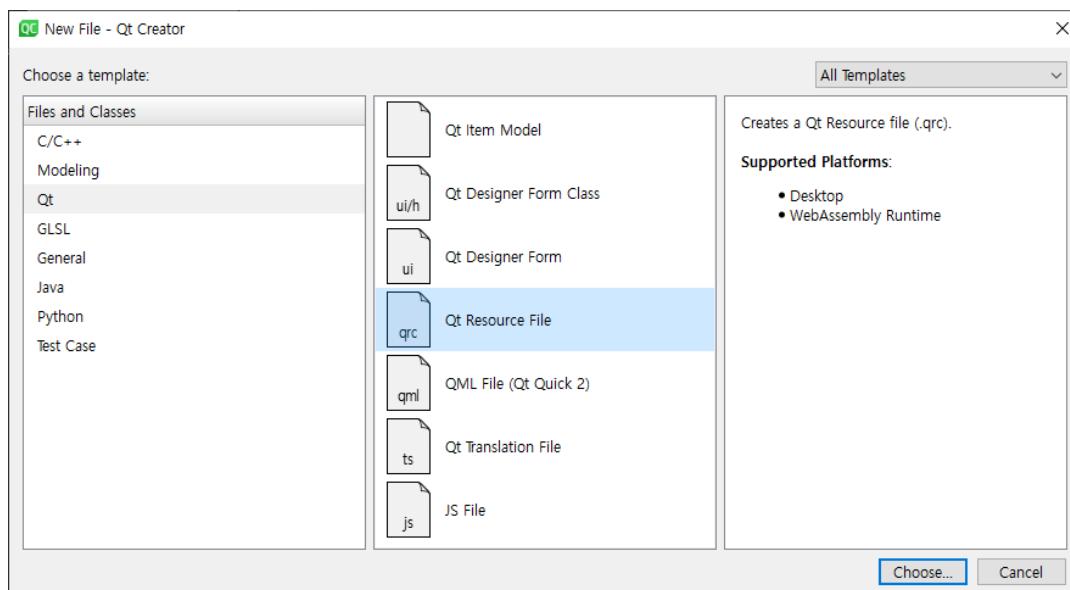


예수님은 당신을 사랑합니다.

Qt Creator창에서 프로젝트를 생성한다. 프로젝트 생성 디아얼로그에서 아래 그림에서 보는 것과 같이 [Generation form] 체크박스를 해제한다.



프로젝트 생성이 완료되면 RESOURCE파일을 추가한다.



Qt Resource File을 추가하고 Scaling 이미지를 추가한다. 이미지는 예제 디렉토리에 있는 이미지를 사용하거나 다른 이미지를 사용해도 된다.

The screenshot shows the Qt Creator interface. The title bar says "widget.h @ 01_ScalingImage - Qt Creator". The left sidebar has icons for Welcome, Edit, Design, Debug, and Projects. The Projects tab is selected. The main area shows a file tree for the "01_ScalingImage" project. Under "Header Files", "widget.h" is selected and highlighted with a red box. Under "Source Files", "main.cpp" and "widget.cpp" are listed. A folder named "resources.qrc" contains a directory structure with "images" and "image.png". The code editor on the right shows the contents of "widget.h".

```

1 ifndef WIDGET_H
2 define WIDGET_H
3
4 include <QWidget>
5
6 class Widget : public
7 {
8     Q_OBJECT
9
10 public:
11     Widget(QWidget *parent = nullptr);
12     ~Widget();
13
14 };
15 #endif // WIDGET_H

```

다음으로 widget.h 헤더파일상에 아래와 같이 paintEvent() 함수를 선언한다.

```

#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();

protected:
    virtual void paintEvent(QPaintEvent *event);

};


```

예수님은 당신을 사랑합니다.

```
#endif // WIDGET_H
```

다음으로 widget.cpp 소스코드 파일을 열고 아래와 같이 소스코드를 작성한다.

```
#include "widget.h"
#include <QPainter>

Widget::Widget(QWidget *parent)
    : QWidget(parent)
{
}

void Widget::paintEvent(QPaintEvent *event)
{
    Q_UNUSED(event)

    QPainter painter;
    painter.begin(this);

    int w = this->window()->width();
    int h = this->window()->height();

    painter.setPen(QColor(0, 0, 0));
    painter.fillRect(0, 0, w, h, Qt::black);

    QPixmap imgPixmap = QPixmap(":/images/image.png")
        .scaled(w, h, Qt::KeepAspectRatio);

    int imgWidth = imgPixmap.width();
    int imgHeight = imgPixmap.height();

    int xPos = 0;
    int yPos = 0;
```

예수님은 당신을 사랑합니다.

```
if(w > imgPixmap.width())
    xPos = (w - imgWidth) / 2;
else if( h > imgPixmap.height())
    yPos = (h - imgHeight) / 2;

painter.drawPixmap(xPos, yPos, imgPixmap);

painter.end();

}

Widget::~Widget() {}
```

위의 소스코드에서 보는 것과 같이 Web Browser의 크기에 맞게 이미지를 Scaling 하기 위해서는 Web Browser의 크기를 얻어와야 한다. Web Browser의 크기를 얻어온 후 QPixmap 클래스에서 제공하는 scaled()함수를 이용해 이미지 크기를 Scaling 할 수 있다.

11. 크로마키 영상 처리 구현

이번 장에서는 이전에서 다루었던 QPainter를 이용해 간단한 Chromakey 응용 어플리케이션을 구현해 보도록 하자.

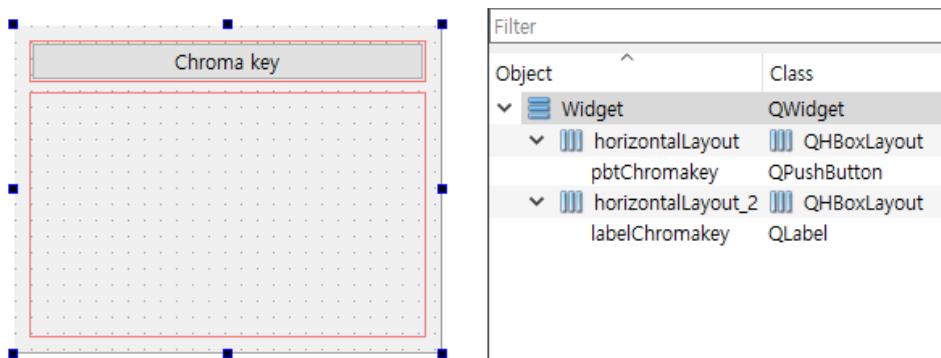
여기서 말하는 Chromakey란 특정 색을 필터링해 다른 이미지의 픽셀로 대체하는 방법을 말한다. 예를 들어 날씨 뉴스 방송의 배경을 다른 이미지나 영상으로 대체하는 영상을 많이 본적이 있을 것이다. 이번 예에서는 사진 이미지 포맷의 배경을 다른 이미지로 변경해 보도록 하자.



위의 그림에서 SOURCE 이미지상에서 배경은 녹색이다. SOURCE 이미지에서 배경색을 TARGET 이미지로 대체한다.

그러기 위해서는 SOURCE 이미지의 사람 부분과 배경을 분리할 수 있는 방법이 필요하다. SOURCE 이미지의 배경은 녹색이다. 그렇기 때문에 사람과 배경을 분리하기 위해서 SOURCE 이미지의 배경색인 녹색의 Threshold 값을 찾으면 사람과 녹색인 배경을 분리할 수 있다. 그리고 분리한 배경을

TARGET 이미지로 변경하면 RESULT 이미지와 같이 된다. QWidget 기반의 새로운 프로젝트를 생성하고 다음과 같이 GUI상에 위젯을 배치한다.



예수님은 당신을 사랑합니다.

위의 그림에서 보는 것과 같이 QPushButton 과 QLabel을 배치한다. 그리고 widget.h 헤더파일에 아래와 같이 소스코드를 작성한다.

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>

QT_BEGIN_NAMESPACE
namespace Ui {
class Widget;
}
QT_END_NAMESPACE

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();

private:
    Ui::Widget *ui;

    QImage m_sourceImg;
    QImage m_targetImg;
    QImage m_resultImg;

    int m_imgWidth;
    int m_imgHeight;
    int m_imgSize;
}
```

예수님은 당신을 사랑합니다.

```
void chromakeyProcess(QImage& sourceImage,
                      QImage& targetImage,
                      QImage& resultImage);

private slots:
    void slotChromakey();

};

#endif // WIDGET_H
```

chromakeyProcess() 함수의 첫 번째 인자는 Chromakey처리를 할 SOURCE IMAGE이다. 두 번째 이미지는 배경 이미지로 녹색을 대체할 이미지이다. 세 번째 인자는 결과를 저장할 것이다. m_imgWidth는 이미지의 가로크기, m_imgHeight는 세로크기 그리고 m_imgSize는 이미지의 크기이다. 여기서 이미지크기 값은 파일의 크기가 아니라 이미지의 가로크기와 세로크기를 곱한다. 그리고 각 픽셀에 RGBA값을 다시 곱한 값이다.

WIDTH * HEIGHT * RGBA(4) = Image Size

이미지 크기의 단위는 Pixel이다. 여기서 주의해야 하는 것 중 하나는 SORUCE 이미지의 크기, TARGET 이미지의 크기 그리고 RESULT 이미지 크기가 모두 동일해야 한다.

다음은 widget.cpp 소스코드 파일이다.

```
#include "widget.h"
#include "./ui_widget.h"

Widget::Widget(QWidget *parent)
    : QWidget(parent)
    , ui(new Ui::Widget)

{
    ui->setupUi(this);
    setWindowTitle("Chromakey Example");

    connect(ui->pbtChromakey, SIGNAL(clicked()),
            this, SLOT(slotChromakey()));

    m_sourceImg = QImage("./images/source.png");
    m_targetImg = QImage("./images/target.png");
```

```
qDebug() << "Image Format : " << m_sourceImg.format();

m_imgWidth = m_sourceImg.width();
m_imgHeight = m_sourceImg.height();
m_imgSize = m_imgWidth * m_imgHeight * 4;

m_resultImg = QImage(m_imgWidth, m_imgHeight, QImage::Format_RGB32);

}

void Widget::chromakeyProcess(QImage& sourceImage,
                             QImage& targetImage,
                             QImage& resultImage)
{
    uchar *pSourceData = sourceImage.bits();
    uchar *pTargetData = targetImage.bits();
    uchar *pResultData = resultImage.bits();

    QColor maskColor = QColor::fromRgb(sourceImage.pixel(1,1));
    int kred      = maskColor.red();
    int kgreen    = maskColor.green();
    int kblue     = maskColor.blue();

    int sPixRed, sPixGreen, sPixBlue;

    for (int inc = 0; inc < m_imgSize ; inc += 4)
    {
        sPixRed    = pSourceData[inc+2];
        sPixGreen = pSourceData[inc+1];
        sPixBlue   = pSourceData[inc];

        if( (abs(kred - sPixRed) +
```

예수님은 당신을 사랑합니다.

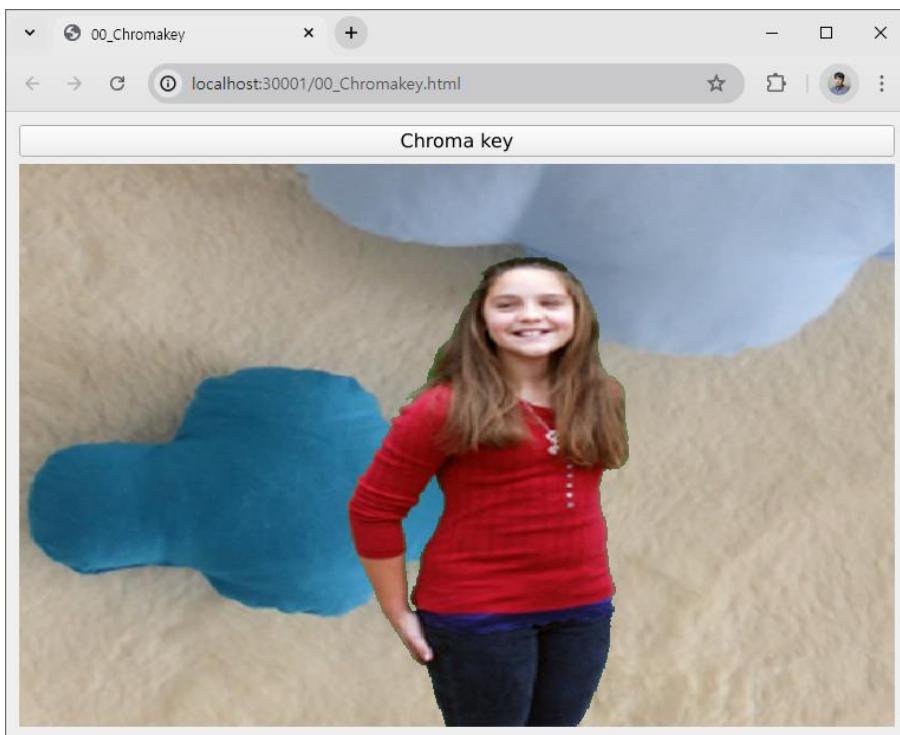
```
abs(kgreen - sPixGreen) +  
abs(kblue - sPixBlue)) / 5 < 22 )  
{  
    pResultData[inc+2] = pTargetData[inc+2];  
    pResultData[inc+1] = pTargetData[inc+1];  
    pResultData[inc] = pTargetData[inc];  
}  
else  
{  
    pResultData[inc+2] = pSourceData[inc+2];  
    pResultData[inc+1] = pSourceData[inc+1];  
    pResultData[inc] = pSourceData[inc];  
}  
}  
}  
  
void Widget::slotChromakey()  
{  
    chromakeyProcess(m_sourcelImg,  
                     m_targetImg,  
                     m_resultImg);  
  
    int width    = ui->labelChromakey->width();  
    int height   = ui->labelChromakey->height();  
  
    QPixmap drawPixmap = QPixmap::fromImage(m_resultImg)  
                      .scaled(width, height);  
  
    ui->labelChromakey->setPixmap(drawPixmap);  
}  
  
Widget::~Widget()
```

예수님은 당신을 사랑합니다.

```
{  
    delete ui;  
}
```

chromakeyProcess() 멤버 함수의 인자로 QImage 를 사용하였다. QPixmap 과 같은 다양한 클래스가 있음에도 불구하고 QImage 클래스를 사용한 이유는 이미지의 각 픽셀의 RGBA 값을 바로 접근할 수 있기 때문이다.

각 픽셀의 RGBA에서 R은 Red, G는 Green, B는 Blue 그리고 A는 Alpha 로 투명 값을 의미한다. 작성한 예제를 실행하고 [RESULT] 버튼을 클릭해 결과를 확인해 보도록 하자.



12. Timer

이번장에서 Qt에서 제공하는 QTimer클래스를 이용해 Timer를 구현해 보도록 하자. QTimer클래스는 지정한 시간을 기준으로 반복해 Slot 함수를 호출할 수 있다.

```
QTimer *timer = new QTimer(this);
connect(timer, SIGNAL(timeout()), this, SLOT(update()));

timer->start(1000);
```

위의 예제 소스코드에서 보는 것과 같이 QTimer 클래스의 오브젝트를 선언한다. 그리고 지정한 시간을 반복해 호출하기 위해서 connect() 함수를 이용해 Signal과 Slot 함수를 연결해야 한다.

connect() 함수에서 첫 번째 인자는 QTimer 오브젝트를 명시한다. 두 번째는 Signal을 명시한다. timeout() Signal 은 지정한 시간이 경과되면 4번째 인자에 지정한 update() Slot 함수가 호출된다.

마지막 라인의 QTimer 클래스의 start() 멤버 함수는 첫 번째 인자로 지정한 시간만큼 경과되면 connect() 함수에서 명시한 Slot 함수가 반복해 호출된다. 이 함수의 첫 번째 인자의 단위는 millisecond 이다.

QTimer 클래스는 stop() 멤버 함수를 이용해 타이머를 정지할 수 있다. 만약 타이머를 반복해서 실행하지 않고 단 한번만 호출되도록 하기 위해 singleShot() 멤버 함수를 사용하면 된다. singleShot() 멤버 함수는 다음 예제 소스코드에서 보는 것과 같이 사용할 수 있다.

```
QTimer::singleShot(200, this, SLOT(updateCaption()));
```

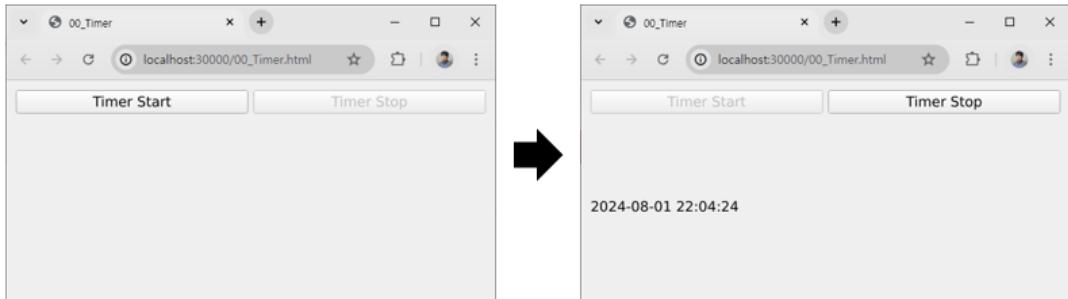
singleShot() 멤버 함수의 첫 번째 인자는 경과 되는 시간이며 단위는 millisecond이다. 그리고 세 번째 인자는 첫 번째 인자의 시간이 경과된 후 호출될 Slot 함수를 지정하면 된다.

- QTimer클래스를 이용한 예제

다음 예제는 1초 간격으로 Slot 함수가 호출된다. 다음 그림에서 보는 것과 같이

예수님은 당신을 사랑합니다.

[Timer Start] 버튼을 클릭하면 타이머가 시작된다. 그리고 [Timer Stop] 버튼을 클릭하면 타이머가 중지된다.



QTimer 예제 실행 화면에서 현재 시간은 QTimer에서 지정한 1000 milliseconds(1초)를 주기로 현재 시간을 시스템으로부터 얻어와 QLabel 위젯에 출력하는 예제이다. 다음 예제 소스코드는 widget.h 소스코드이다.

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>
#include <QTimer>

QT_BEGIN_NAMESPACE
namespace Ui {
class Widget;
}
QT_END_NAMESPACE

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();

private:
```

예수님은 당신을 사랑합니다.

```
Ui::Widget *ui;
 QTimer *m_timer;

public slots:
 void startPressed();
 void stopPressed();
 void elapsedTime();

};

#endif // WIDGET_H
```

startPressed() 는 [Timer Start] 버튼을 클릭하면 호출되는 Slot 함수이다. stopPressed() 함수는 [Timer Stop] 버튼을 클릭하면 호출되는 Slot 함수이다. elapsedTime()함수는 QTimer에서 지정한 시간이 경과되면 호출된다. 다음 예제는 widget.cpp 소스코드이다.

```
#include "widget.h"
#include "./ui_widget.h"
#include <QDateTime>

Widget::Widget(QWidget *parent)
 : QWidget(parent)
 , ui(new Ui::Widget)

{
    ui->setupUi(this);
    setWindowTitle("Timer example");

    connect(ui->pbtStart, &QPushButton::pressed,
            this,           &Widget::startPressed);

    connect(ui->pbtStop,  &QPushButton::pressed,
            this,           &Widget::stopPressed);

    ui->pbtStart->setEnabled(true);
    ui->pbtStop->setEnabled(false);
```

```
m_timer = new QTimer();
connect(m_timer, &QTimer::timeout, this, &Widget::elapsedTime);
}

void Widget::startPressed()
{
    ui->pbtStart->setEnabled(false);
    ui->pbtStop->setEnabled(true);

    m_timer->start(1000);
}

void Widget::stopPressed()
{
    ui->pbtStart->setEnabled(true);
    ui->pbtStop->setEnabled(false);

    m_timer->stop();
}

void Widget::elapsedTime()
{
    QDateTime curr = QDateTime::currentDateTime();
    QString timeStr = curr.toString("yyyy-MM-dd hh:mm:ss");

    ui->leCurrentTime->setText(timeStr);
}

Widget::~Widget()
{
    delete ui;
```

예수님은 당신을 사랑합니다.

}

13. Thread Programming

Qt에서는 Thread를 지원하기 위해서 QThread 클래스를 제공한다. QThread 클래스를 이용해 Thread를 사용하는 경우 특정 상황에서 동기화가 필요한 경우가 있다. 동기화는 Thread 환경에서 다중 사용자가 참조하는 변수를 특정 사용자가 변수 값을 변경할 때 다른 사용자에게 잘못된 값을 참조할 수 있다.

따라서 Thread로 동작되는 특정 함수의 소스코드의 시작 시점부터 종료 시점까지 다른 사용자가 변수를 참조하거나 변경하지 못하게 함으로써 변수가 변경되기 이전의 잘못된 값을 참조하지 못하도록 하기 위한 기능을 제공한다.

Qt에서는 동기화를 지원하기 위해 QMutex 클래스를 제공한다. 다음 예제 소스코드는 QThread 클래스를 상속받아 구현한 클래스의 일부이다.

```
#include <QThread>
#include <QMutex>
#include <QDateTime>

class MyThread : public QThread
{
    Q_OBJECT
public:
    MyThread(int n);
protected:
    void run() override {
        while(!m_threadStop)
        {
            m_mutex.lock();
            ...
            m_mutex.unlock();
            ...
            sleep(1);
        }
    }
};

MyThread::MyThread(int n)
```

```
private:  
    bool m_threadStop;  
    QMutex m_mutex;  
};  
...
```

위의 예제 소스코드에서 보는 것과 같이 QThread를 이용해 MyThread 클래스를 구현한 예제이다. run() 함수는 QThread에서 상속받은 Virtual 멤버 함수이며 Thread에서 동작하고자 하는 기능을 이 함수에서 구현하면 된다.

run() 함수는 내부에서 구현한 함수이기 때문에 클래스 외부에서 QThread에서 제공하는 start() 함수를 호출하면 자동으로 run() 함수가 호출된다.

```
MyThread *thread = new MyThread(this);  
thread->start();
```

간혹 run() 함수를 Public으로 선언하고 외부에서 run() 함수를 외부에서 호출하는 경우가 있다. 이 경우 run() 함수가 Thread에서와 같이 동작하지 않는다.

즉 Thread가 아닌 일반 함수와 같이 동작하므로 run() 함수를 바로 호출하는 경우가 없도록 주의 해야 한다. run() 함수에서 사용한 QMutex 클래스는 동기화를 위한 기능이다.

위의 예제 소스코드에서 보는 것과 같이 QMutex 클래스에서 제공하는 lock() 멤버 함수는 동기화를 시작되도록 선언하고 unlock() 멤버 함수를 통해 동기화의 마지막 구간임을 선언하면 이 구간 내에 선언된 변수는 외부 클래스가 unlock() 멤버 함수 호출될 때 까지 접근할 수 없다.

예를 들어 네트워크 어플리케이션에서 현재 접속자가 10명이라고 가정해 보자. 그리고 현재 접속자가 10명인 경우 값을 users라는 int 공용 변수에 저장했다고 가정해 보자. 만약 새로운 접속자로 인해 11명으로 증가하는 경우, 이 때 users 변수 값을 1증가하는 시점에 동일한 시간 때 현재 접속자를 확인하는 프로세스가 있다면 잘못된 현재 접속자 정보를 다른 접속자에게 잘못된 정보를 줄 수 있다.

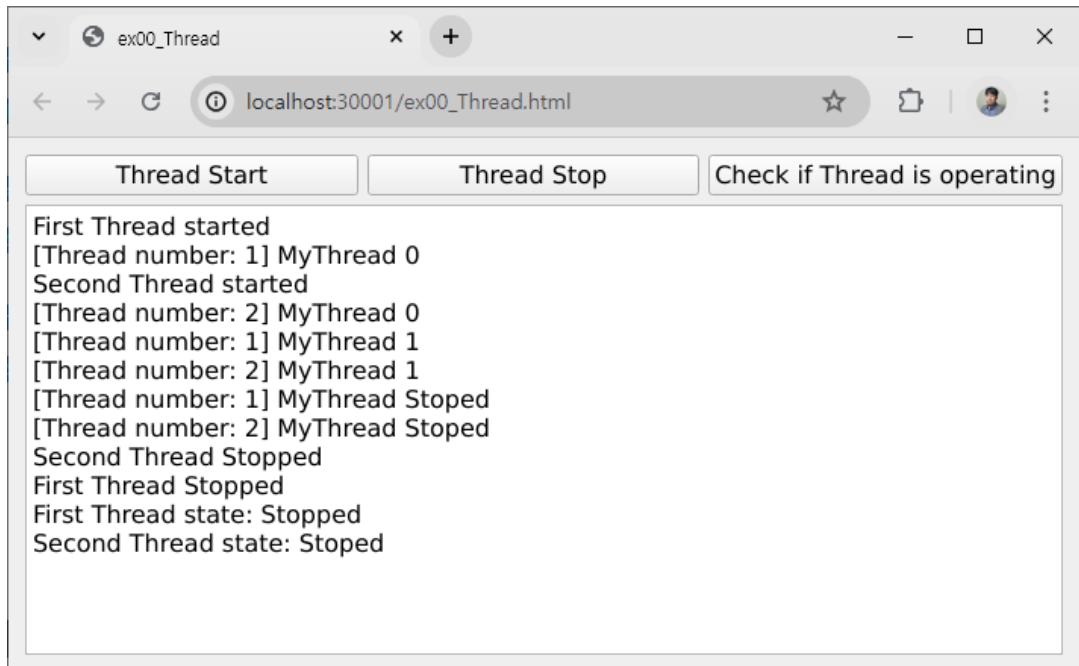
따라서 이럴 때 users 변수 값을 증가하는 소스코드 구간을 lock()과 unlock() 구간에는 users 변수 값을 1 증가 하도록 동기화를 사용하면 users 변수는 변경하거나 참조하지 못하도록 대기 열에서 기다리도록 처리한다.

예수님은 당신을 사랑합니다.

동기화가 필요한 경우 QMutex 클래스를 사용하면 유용하지만 너무 자주 사용하게 되면 프로그램이 Blocking 되어 멈춤 현상이 발생할 수 있으므로 필요한 경우에만 사용하는 것을 권장한다.

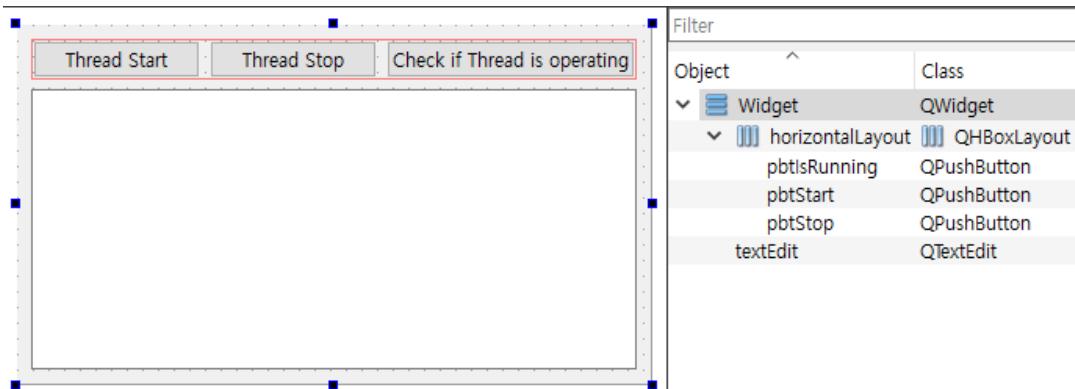
- QThread클래스를 이용한 간단한 예제

이번 예제에서는 아래 그림에서 보는 것과 같이 Web browser상에서 [Thread Start]버튼을 클릭하면 2개의 Thread가 시작된다. 그리고 아래의 그림에서 보는 것과 같이 메시지를 출력한다.



[Thread Stop]버튼을 클릭하면 2개의 Thread가 중지된다. [Check if Thread is operating] 버튼을 클릭하면 2개의 Thread가 동작하는지 확인하는 메시지를 출력한다.

Qt Widget 기반의 프로젝트를 생성하고 아래 그림에서 보는 것과 같이 widget.ui상에 아래와 같이 Widget들을 배치한다.



다음으로 프로젝트상에서 클래스를 추가한다. 클래스의 이름은 MyThread으로 한다. 클래스 생성이 완료되면 MyThread.h 헤더파일을 아래와 같이 작성한다.

```
#ifndef MYTHREAD_H
#define MYTHREAD_H

#include <QThread>
#include <QMutex>

class MyThread : public QThread
{
    Q_OBJECT
public:
    MyThread(int num);
    QString threadMsg();

private:
    bool m_threadStop;
    int m_number;
    QMutex mutex;

public:
    void stop();

signals:

```

예수님은 당신을 사랑합니다.

```
void sig_threadMsg(QString);  
  
protected:  
    virtual void run();  
};  
  
#endif // MYTHREAD_H
```

다음으로 MyThread.cpp 소스코드 파일에 다음과 같이 작성한다.

```
#include "MyThread.h"  
  
MyThread::MyThread(int num)  
{  
    m_number = num;  
}  
  
void MyThread::stop()  
{  
    m_threadStop = true;  
  
    QString str = QString("[Thread number: %1] MyThread Stoped")  
        .arg(m_number);  
    emit sig_threadMsg(str);  
}  
  
void MyThread::run()  
{  
    m_threadStop = false;  
    int i = 0;  
  
    while(!m_threadStop)  
    {
```

예수님은 당신을 사랑합니다.

```
mutex.lock();

QString str = QString("[Thread number: %1] MyThread %2")
    .arg(m_number).arg(i);

i++;
emit sig_threadMsg(str);

mutex.unlock();

sleep(1);
}

}
```

다음으로 widget.h 헤더파일에 아래와 같이 소스코드를 추가한다.

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>
#include "MyThread.h"

QT_BEGIN_NAMESPACE
namespace Ui {
class Widget;
}
QT_END_NAMESPACE

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();
}
```

private:

```
Ui::Widget *ui;  
  
MyThread *m_thread1;  
MyThread *m_thread2;
```

private slots:

```
void slot_pbtStart();  
void slot_pbtStop();  
void slot_isRunning();  
  
void slot_threadMsg(QString);  
  
void slot_thread1_started();  
void slot_thread1_finished();  
void slot_thread2_started();  
void slot_thread2_finished();  
};  
#endif // WIDGET_H
```

다음으로 widget.cpp 소스코드 파일에 아래와 같이 작성한다.

```
#include "widget.h"  
#include "./ui_widget.h"  
#include <QFontDatabase>  
  
Widget::Widget(QWidget *parent)  
    : QWidget(parent)  
    , ui(new Ui::Widget)  
{  
    QFontDatabase::addApplicationFont("./NanumGothic.ttf");  
  
    ui->setupUi(this);
```

```
connect(ui->pbtStart, SIGNAL(pressed()),
        this, SLOT(slot_pbtStart()));
connect(ui->pbtStop, SIGNAL(pressed()),
        this, SLOT(slot_pbtStop()));
connect(ui->pbtIsRunning, SIGNAL(pressed()),
        this, SLOT(slot_isRunning()));

m_thread1 = new MyThread(1);
m_thread2 = new MyThread(2);

connect(m_thread1, SIGNAL(sig_threadMsg(QString)),
        this, SLOT(slot_threadMsg(QString)));
connect(m_thread2, SIGNAL(sig_threadMsg(QString)),
        this, SLOT(slot_threadMsg(QString)));

connect(m_thread1, SIGNAL(started()),
        this, SLOT(slot_thread1_started()));
connect(m_thread1, SIGNAL(finished()),
        this, SLOT(slot_thread1_finished()));

connect(m_thread2, SIGNAL(started()),
        this, SLOT(slot_thread2_started()));
connect(m_thread2, SIGNAL(finished()),
        this, SLOT(slot_thread2_finished()));

}

void Widget::slot_pbtStart()
{
    ui->textEdit->clear();

    m_thread1->start();
```

```
m_thread2->start();
}

void Widget::slot_pbtStop()
{
    m_thread1->stop();
    m_thread2->stop();
}

void Widget::slot_isRunning()
{
    if(m_thread1->isRunning())
        ui->textEdit->append("First Thread state: Running");
    else
        ui->textEdit->append("First Thread state: Stopped");

    if(m_thread2->isRunning())
        ui->textEdit->append("Second Thread state: Running");
    else
        ui->textEdit->append("Second Thread state: Stoped");
}

void Widget::slot_threadMsg(QString msg)
{
    ui->textEdit->append(msg);
}

void Widget::slot_thread1_started()
{
    ui->textEdit->append("First Thread started");
}
```

예수님은 당신을 사랑합니다.

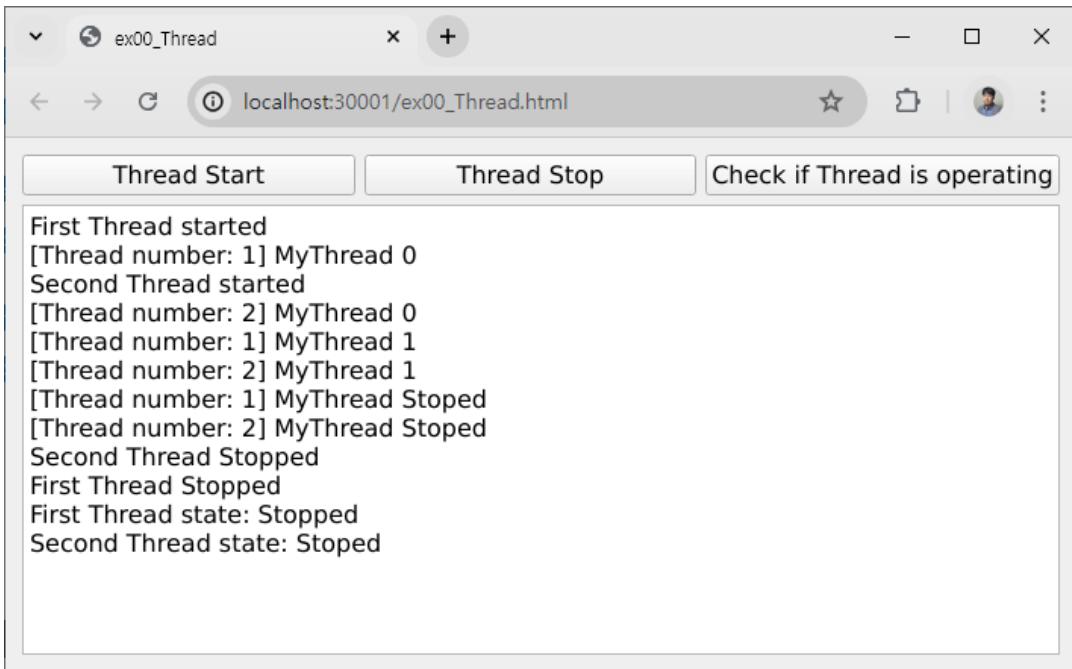
```
void Widget::slot_thread1_finished()
{
    ui->textEdit->append("First Thread Stopped");
}

void Widget::slot_thread2_started()
{
    ui->textEdit->append("Second Thread started");
}

void Widget::slot_thread2_finished()
{
    ui->textEdit->append("Second Thread Stopped");
}

Widget::~Widget()
{
    delete ui;
}
```

위와 같이 작성하고 빌드한 다음, 실행하면 Web browser상에서 아래와 같이 실행되는 것을 확인할 수 있다.



- Reentrancy 와 Thread-Safety를 만족하는 Thread 예제

Reentrancy는 두 개 이상의 Thread가 동작되고 있을 때 Thread 가 수행되는 순서에 상관없이 하나의 Thread 가 수행되고 난 후, 다음 Thread 가 수행된 것처럼 수행되는 것을 의미한다.

Thread-Safety는 두 개 이상의 쓰레드가 동작하는 상황에서 Static 또는 Heap 메모리 영역과 같이 공유되는 메모리 영역을 접근할 때 안정성을 보장하기 위해 Mutex 와 같은 메커니즘을 사용하는 것을 의미한다. 다음 소스코드 두 개의 Thread를 생성하고 Reentrancy 와 Thread-Safety를 고려하지 않은 Thread를 구현한 예제이다.

```
static QMutex mutex;  
static QWaitCondition incNumber;  
static int numUsed;  
  
class Producer : public QThread  
{  
    Q_OBJECT  
public:  
    void run() override {  
        for(int i = 0 ; i < 10 ; i++) {
```

예수님은 당신을 사랑합니다.

```
sleep(1);
++numUsed;
}

}

public:
Producer() {}

};

class Consumer : public QThread
{
Q_OBJECT
void run() override {
for(int i = 0 ; i < 10 ; i++) {
qDebug("[Consumer] numUsed : %d", numUsed);
}
}

public:
Consumer() {}
};
```

위의 예제 소스코드에서 보는 것과 같이 Producer 와 Consumer 클래스를 구현하고 main.cpp 에서 다음과 같이 작성해 보도록 하자.

```
#include <QCoreApplication>
#include "mythread.h"

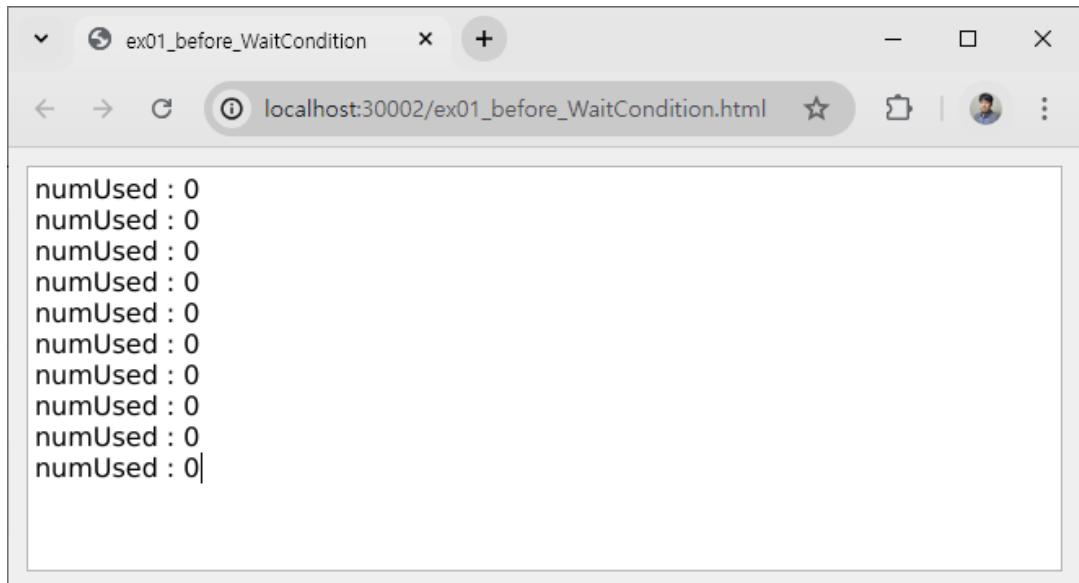
int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);

    Producer producer;
    Consumer consumer;
    producer.start();
```

```
consumer.start();

return a.exec();
}
```

위의 예제 소스코드에서 2개의 구현한 Thread 클래스 오브젝트를 선언하고 QThread에서 제공하는 start() 멤버 함수를 호출하면 다음 그림에서 보는 것과 같이 Thread를 시작한다.



예제 실행 화면에서 보는 것과 같이 Consumer 클래스의 run() 함수가 먼저 모두 수행된 후 Producer 클래스의 run() 함수가 수행될 것이다.

다음 예제는 Reentrancy 와 Thread-Safety를 만족하는 Thread를 구현한 예제이다. 이전 예제에서 구현한 Consumer 클래스와 Producer 클래스를 다음과 같이 수정해 보도록 하자.

```
#ifndef MYTHREAD_H
#define MYTHREAD_H
#include <QWaitCondition>
#include <QMutex>
#include <QThread>

static QMutex mutex;
```

```
static QWaitCondition incNumber;
static int numUsed;

class Producer : public QThread
{
    Q_OBJECT
public:
    void run() override {
        for(int i = 0 ; i < 10 ; i++) {
            sleep(1);
            ++numUsed;
            qDebug("[Producer] numUsed : %d", numUsed);
            mutex.lock();
            incNumber.wakeAll();
            mutex.unlock();
        }
    }
};

class Consumer : public QThread
{
    Q_OBJECT
public:
    void run() override {
        for(int i = 0 ; i < 10 ; i++) {
            mutex.lock();
            incNumber.wait(&mutex);
            mutex.unlock();
            qDebug("[Consumer] numUsed : %d", numUsed);
        }
    }
};
```

```
Consumer() { }  
};  
  
#endif // MYTHREAD_H
```

이전의 예제 소스코드에 작성했던 Producer 와 Consumer 클래스에서는 두 Thread 간의 실행과 관계없이 Thread 가 동작한다. 하지만 수정한 Producer 와 Consumer 클래스는 Reentrancy 와 Thread-Safety를 만족하는 Thread 클래스를 구현한 예제이다.

Producer 와 Consumer Thread는 numUsed 값이 변경될 때에 동작한다. 즉 Consumer는 QWaitCondition 클래스의 wait() 멤버 함수에 의해 Waiting 되어 있는 상태이다.

Producer 클래스가 numUsed 값을 1 증가시키고 QWaitCondition 클래스의 wakeAll() 멤버 함수를 이용해 Consumer 클래스를 Wakeup 한다.

이러한 방식으로 두 개 이상의 Thread에서 Reentrancy 와 Thread-Safety를 만족하는 Thread를 쉽게 구현할 수 있다.

